SEQUENTIAL QUADRATIC PROGRAMMING METHODS BASED ON INDEFINITE HESSIAN APPROXIMATIONS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF OPERATIONS RESEARCH
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Meredith J. Goldsmith
March 1999

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Walter Murray
(Principal Adviser)

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Michael A. Saunders

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Richard W. Cottle

Approved for the University Committee on Graduate Studies:

Abstract

We address the nonlinearly constrained optimization problem: to find a local minimizer for an objective function subject to nonlinear inequality constraints, where all functions are twice continuously differentiable. A key challenge in developing a fast algorithm for solving this problem is to find an accurate approximation to the Hessian of the Lagrangian $\nabla_x^2 \mathcal{L}(x, \lambda)$, the second-derivative matrix that reflects the curvature of the objective and constraints. In this dissertation we develop a new technique for forming a quasi-Newton approximation to the Hessian of the Lagrangian and apply it to a sequential quadratic programming (SQP) method.

The common approach in approximating $\nabla_x^2 \mathcal{L}(x, \lambda)$ has been to follow quasi-Newton techniques used for unconstrained optimization. A single, positive-definite approximation matrix is maintained using the BFGS quasi-Newton update, and typically the dependence of the Lagrangian on the multipliers is ignored. This *direct approximation* may be poor even on ideal problems. We aim to improve the quality of the quadratic model and the convergence rate of the SQP algorithm with the following alternative approach. We first approximate the Hessian of the objective function and the Hessian of each constraint function using the symmetric rank-one (SR1) update, and then we combine these individual estimates to obtain an estimate of $\nabla_x^2 \mathcal{L}(x, \lambda)$. This *disaggregated approximation* can be applied efficiently to large, sparse problems.

An important challenge in this technique is that it can result in an indefinite approximation, which when used within an SQP method creates a nonconvex subproblem. When the direct approximation is used, the QP subproblems that arise in SQP algorithms are strictly convex and have unique solutions. An extensive theory of SQP methods is based on the property that the solution of the QP is a descent direction for various merit functions. When the QP is not convex, this property may not hold. To obtain a descent direction may require modifying the QP subproblem. Our objective is to construct a suitable search

direction without modifying the subproblem unnecessarily.

Murray and Prieto's second-derivative SQP algorithm [MP99] uses an indefinite Hessian approximation, but constructs search directions based on terminating QP subproblems (even convex ones) at the first stationary point. We develop rules and Hessian modifications that eliminate the need to stop at the first stationary point. If the QP is strictly convex, the rules enable the minimizer to be determined without any modification to the subproblem. A MATLAB implementation of this SQP algorithm with the disaggregated Hessian approximation and the SR1 quasi-Newton update achieves superior performance on certain problems when compared to the direct BFGS Hessian approximation.

Acknowledgements

Many people deserve credit for their ideas and support that led me to complete this dissertation, but one person stands out—my advisor, Professor Walter Murray. He gave me tremendous help, inspiration, and guidance during my time at Stanford. It has been an honor to learn from someone of such great expertise and insight and a pleasure to work with someone so good-humored and patient. He has been generous with his ideas and his time as well as his opinions. His high standards brought me to achieve work of high quality.

Many thanks are due to Professor Michael A. Saunders and Professor Richard W. Cottle for serving on my oral and reading committees. I greatly appreciate Professor Saunders' patience and care in going over drafts of my dissertation in great detail. He suggested many improvements and helped me to improve the clarity and presentation of this dissertation. Professor Richard W. Cottle diligently read this document in a timely manner and provided many helpful comments, despite other pressing responsibilities. I thank Professor B. Curtis Eaves and Professor Stephen P. Boyd for serving on my orals committee.

I am obliged as well to Professor Francisco J. Prieto, who laid the foundations for my research work on sequential quadratic programming methods when he was a Ph.D. student at Stanford. My MATLAB code is based on a program he wrote, and he spent a few weeks helping me to understand and modify the program.

I would like to thank all the faculty and staff of the Department of Engineering-Economic Systems and Operations Research for their help and advice over the years. Financial support for this research was provided by the Systems Optimization Laboratory at Stanford, the National Science Foundation, the Office of Naval Research, and by the Department of Operations Research through a first-year fellowship.

Many friends—too many to name here—made my time at Stanford enjoyable, and I thank all of them for their support. I am indebted to my boyfriend Sergey Brin, clip-art expert extraordinaire, for helping me create presentations of this work as well as for his encouragement, interest, and editorial assistance.

Last, but not least, this dissertation is dedicated to my parents Barbara and Richard and my sister Rebecca. Their love, advice, and encouragement prepared me well for the difficult task of earning a Ph.D.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The problem of finding a local minimizer $x \in I\!R^n$ for a nonlinear function $f(x)$ subject to a set of nonlinear constraints $c(x) \geq 0$, where $c(x) \in I\!R^m$, is the *nonlinearly constrained optimization problem* NP. There are many alternative forms of this problem. Initially, we will focus on the *inequality constrained nonlinear program*

$$\text{INP} \qquad \begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & c(x) \geq 0, \end{array}$$

where $f(x)$ and $c_i(x)$ are assumed to be twice continuously differentiable functions. Later, and in particular for subproblems, we will convert to the *equality and bound constrained nonlinear program*

$$\text{BNP} \qquad \begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & c(x) = 0, \quad x \geq 0. \end{array}$$

Nonlinear optimization problems have applications in chemical processing, trajectory optimization, mechanical structure design, and many other areas. Optimization problems with nonlinear constraints are significantly more difficult to solve than unconstrained or linearly constrained optimization problems. Algorithms to solve INP may take many iterations and function evaluations, with each function evaluation being an expensive operation in some cases. Performance in practice can be measured by counting iterations, function evaluations, gradient evaluations, and the computational cost of determining an iterate.

Sequential quadratic programming (SQP) is a popular class of methods considered to be effective and reliable for solving INP. At each iteration of an SQP method, one solves a quadratic program (QP) that models INP at the current iterate. The solution to the QP is used as a search direction to determine the next iterate. The QP objective is usually assumed to be strictly convex.

In this thesis, we introduce a quasi-Newton approximation that produces a nonconvex QP subproblem[1], and we describe an SQP method that allows for such a subproblem. In certain cases, the approach advocated performs better than the standard positive-definite approximation. The quality of the quadratic model is better, and this impacts the efficiency and the convergence rate of the method.

This chapter begins by introducing the problem INP, defining what we mean by a "solution," and summarizing the common features of methods for solving INP. The rest of the chapter gives background on SQP methods, including local and global convergence properties, quasi-Newton Hessian approximations, and the QP subproblem. Finally there is a preview of later chapters.

1.1 Properties of a solution to INP

Local minimizer

A *neighborhood* $N(x^*)$ is defined as the set of all x such that $\|x - x^*\| < \delta$, for some $\delta > 0$.

Definition 1.1.1 *A point x is a local minimizer of INP if*

1. x^* is feasible with respect to all the constraints, i.e., $c(x^*) \geq 0$;

2. there exists a neighborhood $N(x^*)$ such that

$$f(x^*) \leq f(x) \text{ for all feasible } x \in N(x^*). \tag{1.1}$$

Methods for solving INP are based on seeking a point satisfying conditions that hold at a local minimizer. Fundamental to the understanding of these conditions is the Lagrangian, a function in the variables x and the Lagrange multipliers λ, defined as

$$\mathcal{L}(x, \lambda) \equiv f(x) - \lambda^T c(x).$$

[1] "QP subproblem that might not be strictly convex" is more accurate, but the correct meaning should be clear from context.

The Lagrangian is used to express first-order and second-order *optimality conditions* for a local minimizer. The first-order optimality conditions are also known as the *Karush-Kuhn-Tucker (KKT) conditions.*

Optimality conditions

The term *active constraint* will be used to designate a constraint that is satisfied exactly at the current point, and the set of all constraints active at a given point will be referred to as the *active set* at the point. Let $\widehat{c}(x)$ be the vector of *active constraints* at x, and let $\widehat{A}(x)$ be the matrix whose row i is the transposed gradient vector $\widehat{c}_i(x)^T$.

A *constraint qualification* must hold to permit the analysis of feasible arcs required by the optimality conditions. One such constraint qualification is that the gradients of the active constraints at x^* be linearly independent, i.e., the matrix $\widehat{A}(x^*)$ should have full row rank. A point that satisfies this particular constraint qualification is known as a *regular point.* For more background on the optimality conditions, see [BSS93, pp. 131–195].

If x^* is a local minimizer and a regular point of INP, then for some Lagrange multiplier $\lambda^* \geq 0$,

$$c(x^*) \geq 0, \qquad c(x^*)^T\lambda^* = 0, \qquad \nabla_x \mathcal{L}(x^*, \lambda^*) = \nabla f(x^*) - A(x^*)^T\lambda^* = 0, \tag{1.2}$$

where $A(x) = \nabla c(x)$ is the Jacobian matrix. These conditions are known as the *first-order necessary optimality conditions*, and a point that satisfies these conditions is known as a *KKT point.* Note that x^* is a stationary point of the Lagrangian (with respect to x), but not necessarily an unconstrained minimizer of the Lagrangian. For the special case of *convex programs,* $\nabla^2 f(x^*) \succ 0$[2].

Suppose x^* is a local minimizer and a regular point of INP. Then, x^* is a KKT point and

$$\forall\, v \in \mathcal{N}(A^*) \qquad v^T \nabla_x^2 \mathcal{L}(x^*, \lambda^*)\, v \geq 0, \tag{1.3}$$

where $\mathcal{N}(A^*)$ denotes the null-space of $A^* = A(x^*)$. These are known as the *second-order necessary optimality conditions.* Unless x^* is a vertex, second derivatives are required to determine if x^* satisfies (1.3). Since the algorithms we explore require only first derivatives, we cannot hope to prove such algorithms terminate at a point satisfying (1.3), and so we

[2]The symbol $\succ$ is used to indicate a matrix is positive definite, and $\succeq$ indicates positive semi-definiteness.

consider a "solution" to be any KKT point. We use the term *stationary point* for a point x that is feasible and for which λ exists such that $\nabla_x \mathcal{L}(x, \lambda) = 0$.

1.2 Common features of methods

Measuring algorithmic performance

Successful algorithms for INP need to be both efficient and robust. Under reasonable assumptions, a robust algorithm must be *globally convergent* (convergent from any starting point), and able to solve in practice both well-conditioned and ill-conditioned problems. The ability of an algorithm to achieve a high level of accuracy in the solution also depends on its convergence rate in the neighborhood of the solution. For small dense problems, the efficiency of an algorithm can be measured by counting operations such as function evaluations and gradient evaluations. For sparse problems, storage requirements and the cost of operations, such as factorizing a matrix or matrix-vector multiplication, must also be considered, and computation time is frequently reported. Analysis of this type has been applied to large-scale algorithms such as MINOS [MS78, MS82] and SNOPT [GMS97].

Techniques for global convergence

Linesearch methods Nearly all techniques for nonlinear programming are iterative, producing a sequence of subproblems related in some way to the original problem. Certain methods in their elementary form (such as Newton methods) have rapid local convergence rates, but may fail to converge from all starting points. *Linesearch* methods are one means of ensuring global convergence while attempting to maintain fast local convergence.

Linesearch methods limit the size of the step taken from the current point to the next iterate. Such methods generate a sequence of iterates of the form

$$\bar{x} = x + \alpha p, \tag{1.4}$$

where p is the search direction obtained from the subproblem, and α is a positive scalar *steplength*. For unconstrained minimization, or if feasibility is maintained for the constraints, the best steplength is one that minimizes the objective function $f(\bar{x})$. However, determining a minimizer along p is also an iterative process, and this time-consuming operation is typically not done in practice. Instead, x is determined by a finite process that ensures a sufficient reduction in $f(x)$. For an overview of linesearch methods (for unconstrained optimization), see Chapter 1 of Fletcher [Fle87].

Trust-region methods Linesearch methods are our focus; however, the results obtained for quasi-Newton approximations are equally applicable to *trust-region* methods, the main alternative to linesearch methods. The motivation behind the trust-region approach is that the minimum of the local quadratic model should be accepted so long as the model adequately reflects the behavior of the function(s) under consideration. Trust-region methods choose a radius Δ and then determine $\bar{x}$ that is the global minimizer of a model of the function subject to $\|\bar{x} - x\| \leq \Delta$.

For unconstrained problems, if $f(\bar{x})$ is sufficiently lower than $f(x)$ (compared to the change predicted by the model), Δ is reduced according to some rule, and the process is repeated. If the reduction in $f(\bar{x})$ is acceptable, Δ can be made larger. For an overview of trust-region methods (for unconstrained optimization), see Chapter 5 of Fletcher [Fle87].

Merit functions Regardless of whether a linesearch or trust-region method is used, when feasibility of the iterates is not maintained, it can be difficult to guide the choice of steplength. For problems with only linear constraints, it is straightforward to maintain feasibility at every iteration. But, when even a single constraint is nonlinear, maintaining feasibility at every iteration becomes difficult, if not impossible. For infeasible iterates, it is not immediately obvious how to choose a steplength α_k; we would like the next iterate to minimize the objective function, but we might also like to reduce the infeasibilities of the constraints. To ensure that progress is made towards the solution we define a *merit function* that can be used to measure whether one point is better than another.

Quadratic model of Lagrangian

A wide variety of methods have been proposed for solving constrained optimization problems; these include reduced-gradient methods (attributed to Wolfe [Wol62]), penalty and barrier function methods (see [FM68]), methods based on augmented Lagrangians [AS58, Hes69, Pow69] or projected augmented Lagrangians [Rob72], and SQP methods. (For a survey on constrained optimization methods, refer to the chapter by Gill *et al.* in the optimization volume edited by Nemhauser *et al.* [GMSW89, pp. 171–210].) All of these methods form quadratic approximations to composite functions of the objective function and constraints, this being the Lagrangian in the case of SQP methods. Quasi-Newton methods are used to approximate the Hessians.

Quasi-Newton methods were originally developed for unconstrained optimization, where steps are taken that minimize a quadratic model of a single function, the objective function. A key feature of quasi-Newton methods is that the Hessian approximation is based on first derivatives only. For linesearch methods, the Hessian approximation needs to be positive definite, but trust-region methods allow the matrix to be indefinite. Quasi-Newton techniques have been extended directly to constrained optimization, despite the structure of the Hessian being more complex. (For a constrained problem, the Hessian of the quadratic model incorporates the objective function and the constraints, and often includes Lagrange multiplier estimates or penalty parameters.) Typically the composite Hessian is approximated using a single matrix and the BFGS quasi-Newton update.

We propose to *disaggregate* the Hessian of the Lagrangian and form a separate quasi-Newton approximation for the Hessian of each constraint function and the objective function. While the BFGS update is the clear choice for positive-definite Hessian approximations in unconstrained optimization, in the constrained case using the symmetric rank-one (SR1) update has advantages. Along with the SR1 update, disaggregation is used to improve the properties of the Hessian approximation.

Disaggregating the quasi-Newton Hessian approximation has benefits not only for SQP, but for other methods as well. For example, augmented Lagrangian methods have subproblems that minimize the Lagrangian plus an additional quadratic penalty term in the constraints, and such subproblems may be solved using a quasi-Newton method. Disaggregating the Hessian approximation allows easy incorporation of exact second derivatives for components of the Hessian that can be computed analytically. Moreover, quasi-Newton estimates from one iteration can be used as a starting point for the next major iteration.

On the other hand, disaggregation can make the Hessian approximation indefinite and may require significantly more storage. For trust-region methods, indefiniteness does not present a problem. For penalty-function methods, the indefinite Hessian could be modified to be positive definite by a modified Cholesky factorization, executed once per iteration. But for SQP methods, it is necessary to revise both the QP subproblem and the method of solution.

1.3 SQP methods

1.3.1 Background

The original SQP method due to Wilson [Wil63] generates a sequence of directions p, each of which is the minimizer to a QP subproblem that is a local model of INP. This is an iterative method from some starting point x_0, and at iteration k, the basic SQP method takes the full step $x_{k+1} \leftarrow x_k + p$. For simplicity, we omit the subscript k whenever clear from context. In general, $x \equiv x_k$ and $\bar{x} \equiv x_{k+1}$.

The QP subproblem directly corresponding to INP has the form

IQP	$\displaystyle\underset{p}{\text{minimize}} \quad g^T p + \tfrac{1}{2} p^T H p$
	subject to $\; c + Ap \geq 0,$

where $g = g(x) = \nabla f(x)$ is the gradient of the objective at the current iterate x, $H = \nabla^2_{xx} \mathcal{L}(x, \lambda)$ is the Hessian of the Lagrangian with latest multiplier estimate λ, $A = A(x)$ is the Jacobian of the constraints, and $c = c(x)$. It is often assumed that $H \succ 0$, in which case the QP has a unique solution.

The linear constraints of IQP are a first-order approximation to the nonlinear constraints at the current iterate x. The quadratic objective of the QP models the curvature of the Lagrangian. For reasons to be discussed later, the QP gradient g equals the objective gradient rather than the full gradient of the Lagrangian.

In the neighborhood of a solution and under certain other assumptions, the solution x^* to INP minimizes the quadratic model to the Lagrangian $\mathcal{L}(x, \lambda^*)$ in the subspace defined by the linearized active constraints at x^*. In other words, x^* is the solution to the IQP subproblem at $x = x^*$ and $\lambda = \lambda^*$.

In its original form, Wilson's SQP method is neither easy to implement (it requires exact second derivatives), nor guaranteed to converge, for the same reasons Newton's method for unconstrained optimization may fail to converge. Murray [Mur69] suggested a response to the first problem, that the Hessian of the Lagrangian could be replaced by a quasi-Newton approximation B. To ensure *global* as well as local convergence, we choose to analyze SQP methods combined with a linesearch method (1.4), having updates of the form $\bar{x} = x + \alpha p$, where the steplength α is determined by a linesearch to reduce a merit function.

For an overview of SQP methods, see [GMW81] or Powell [Pow83]. For recent work on large-scale SQP, see [GMS97] or [Mur97].

1.3.2 Local and global convergence properties

In practice, SQP methods have proven to be extremely effective on general nonlinearly constrained problems. Favorable local convergence properties of SQP follow, under suitable assumptions, from the fact that when B closely approximates $\nabla_x^2 \mathcal{L}(x, \lambda)$, the QP subproblems generate the Newton direction. If unit steps are taken and the method converges, a quadratic asymptotic rate of convergence ensues. With a carefully chosen merit function, the method is globally convergent with no deterioration in the convergence rate.

Merit functions

Merit functions are used to define an "improving" sequence and to prevent the method from diverging even from unfavorable starting points. If a merit function is used in conjunction with a linesearch to choose the steplength α at each iteration, an SQP method can be proven to converge globally.

Three important considerations when choosing an appropriate merit function for an algorithm are

1. Choosing steplengths based on the merit function should lead to global convergence.

2. It should be possible to achieve sufficient decrease in the merit function given the search direction defined by the subproblem (i.e., for some value of the steplength).

3. The merit function should not inhibit the rate of convergence (i.e., the steplength should be 1 whenever possible).

Popular choices for the merit function with the SQP method include the *absolute value merit function* and the *augmented Lagrangian merit function*. Both of these merit functions use penalty parameters, and the choice of penalty parameter can have a substantial effect on efficiency.

The absolute value (or ℓ_1) merit function is

$$M_1(x, \rho) \equiv f(x) + \rho \sum_i |c_i^-(x)|, \tag{1.5}$$

where

$$c_i^- \equiv -\min(0, c_i).$$

It can be shown that if the penalty parameter ρ is sufficiently large, a local minimizer of INP is also a local minimizer of (1.5). Like the above function, most merit functions involve a parameter whose value is not known initially (e.g., ρ). The absolute value merit function is an *exact penalty function*, in the sense that for ρ sufficiently large, x^* is an unconstrained minimizer of $M_1(x, \rho)$. In addition, ρ can always be chosen so that the SQP search direction p from the QP subproblem is a descent direction. However, requiring a decrease in M_1 can inhibit superlinear convergence (the "Maratos" effect).

The augmented Lagrangian merit functions are based on the fact, implied by the optimality conditions, that x^* is a stationary point of the Lagrangian (using optimal multipliers) but not necessarily a minimizer of the Lagrangian. An *augmented Lagrangian* is created by augmenting the Lagrangian with a new term that vanishes at x^*, but alters the Hessian of the Lagrangian to make it positive definite in the subspace of vectors defined by $\widehat{A}(x^*)$.

We shall be concerned mainly with a special augmented Lagrangian merit function $L_A(x, \lambda, s, \rho)$ that has continuous second derivatives; the Lagrange multipliers and constraint slacks are treated as variables that are updated during the linesearch. More will be said about this merit function in Chapter 2.

To ensure global convergence, the search direction found from the QP is required to be a direction of sufficient descent for the chosen merit function. For a positive-definite QP (a QP with a positive-definite Hessian), the solution to the QP provides this descent direction because, for a sufficiently large penalty parameter ρ, it can be shown that the following inequality can always be satisfied:

$$(\Delta x^T, \Delta\lambda^T, \Delta s^T)\nabla L_A \leq -\frac{1}{2}\Delta x^T B \Delta x. \tag{1.6}$$

An important point is that this inequality is satisfied for *any* positive value of ρ when $\|c^-\|$ is sufficiently small (and $B \succ 0$).

Local convergence with exact second derivatives

Given an equality-constrained nonlinear problem ENP and a point x_0, suppose exact second derivatives and multiplier estimates are available, H is set to $\nabla^2_{xx}\mathcal{L}(x_0, \lambda_0)$, and $H \succ 0$. Then, the direction p that is the solution of the QP subproblem produced by the SQP method is the same as the "Newton step" $\bar{p}$ that is the step computed by Newton's method applied to the first-order necessary conditions for ENP.

The first-order necessary conditions for ENP that hold at a minimizer (x^*, λ^*) are

$$\lambda^* \geq 0, \qquad \nabla_x \mathcal{L}(x^*, \lambda^*) = 0, \qquad c(x^*) = 0, \tag{1.7}$$

where $g(x) \equiv \nabla f(x)$, $A(x) \equiv \nabla c(x)$, and $\nabla_x \mathcal{L}(x^*, \lambda^*) = g(x^*) - A(x^*)^T \lambda^*$. The Newton step $(\bar{p}, \bar{\lambda})$ from (x_0, λ_0) towards a KKT point (x^*, λ^*) (a point that satisfies (1.7)) is the solution to the system of equations

$$\begin{pmatrix} \nabla^2_{xx} \mathcal{L}(x_0, \lambda_0) & A(x_0)^T \\ A(x_0) & 0 \end{pmatrix} \begin{pmatrix} \bar{p} \\ -\bar{\lambda} \end{pmatrix} = - \begin{pmatrix} g(x_0) - A(x_0)^T \lambda \\ c(x_0) \end{pmatrix}. \tag{1.8}$$

The first-order necessary conditions for a solution (p, λ^*) to the QP subproblem,

EQP		
	$\underset{p}{\text{minimize}}$	$g(x_0)^T p + \frac{1}{2} p^T \nabla^2_{xx} \mathcal{L}(x_0, \lambda_0) p$
	subject to	$c(x_0) + A(x_0) p = 0,$

are equivalent to (1.8) except that λ in the QP is an estimate of λ^* itself rather than an estimate of $\lambda^* - \lambda_0$. It follows that $p = \bar{p}$, and hence the rate of convergence of the SQP method under the above assumptions (exact second derivatives, $H \succ 0$, unit steps taken) should be the same as for Newton's method. Under mild additional assumptions, Newton's method has a quadratic asymptotic rate of convergence. This analysis generalizes to the inequality case following a result by Robinson [Rob72] that shows the active set at the solution of the QP is identical to that of the nonlinear program when sufficiently close to the solution.

For more details on the quadratic convergence of SQP methods, see Fletcher [Fle74, pp. 228–234] or Bertsekas [Ber82, pp. 234–256]. Goodman [Goo85] also proves quadratic convergence for an algorithm derived from Newton's method for minimizing nonlinear equality constraints and shown to be equivalent to SQP with least-squares Lagrange multiplier estimates.

Local convergence with approximate Hessian

When exact second derivatives of the objective function f and the constraints c are available, the ideal choice for the QP Hessian at the kth iteration is $\nabla^2\mathcal{L}(x_k, \lambda_k)$, where x_k and λ_k are the latest approximations to x^* and λ^*.

If approximations are used in place of exact second derivatives, the steps taken are no longer Newton steps, and the convergence rate may deteriorate. Let B_k be an approximation to the Hessian of the Lagrangian at the kth iteration. If it is assumed, as by Boggs, Tolle, and Wang [BTW82], that the $\{B_k\}$ satisfy

$$\lim_{k\to\infty} \frac{\|Z_k^T(B_k - \nabla_{xx}^2\mathcal{L}(x^*, \lambda^*))p_k\|}{\|p_k\|} = 0, \tag{1.9}$$

then superlinear convergence can be shown for the SQP method, under certain other assumptions.

When an inexact Hessian approximation is used, the infeasibilities tend to converge to zero faster than the reduced gradient converges to zero (that is, feasibility is reached faster than optimality). This happens because the component of the search direction p in the null-space of the constraints (the optimality component) is based on *approximate* second-derivative information, whereas the range-space component of p (and hence the degree of infeasibility) is much less dependent on H. If unit steps are eventually taken and the Jacobian of the active constraints at the solution has full rank, then the infeasibilities (and hence the range-space component of p) converge to zero at a quadratic rate. Consequently, final search directions tend to have null-space components that are large relative to the range-space components. Sometimes the unit step is not taken because either the Hessian approximation is poor or the Jacobian at the solution is singular (or ill-conditioned). Under such circumstances, the range-space and null-space components of p_k converge linearly at best.

1.3.3 Approximating the Hessian

To assure a unique minimizer for the subproblem, some SQP methods *require* the QP Hessian H to be positive definite. Since the Hessian of the Lagrangian can be indefinite even at the solution, this requirement interferes with the effective use of exact second derivatives. A further difficulty with exact second derivatives is that in many cases they are either unknown or too costly to compute.

In practice the Hessian of the Lagrangian is often approximated from first derivatives using a quasi-Newton approach that produces a positive-definite matrix. Most often a *single* symmetric matrix is updated at each iteration. We refer to this approach as the *direct approximation* of the Hessian.

Quasi-Newton methods for unconstrained optimization

Quasi-Newton methods were originally developed for unconstrained and linearly constrained problems. They are "Newton-like" methods in which the Hessian of a function, say $\mathcal{F}$, is replaced by an approximation matrix B. This approximation is usually based on first-derivative information of $\mathcal{F}$. Beginning with an initial matrix such as the identity matrix, it is updated at each iterate x in the form

$$\bar{B} = B + E. \tag{1.10}$$

The intent is that eventually the matrix B will become a close approximation of the exact Hessian. At every iteration, a quasi-Newton method attempts to update the approximation matrix with curvature information gained along the most recent search direction. This is accomplished by choosing an update E that satisfies the *quasi-Newton condition*,

$$\bar{B}(\bar{x} - x) = \nabla\bar{\mathcal{F}} - \nabla\mathcal{F}. \tag{1.11}$$

Define

$$\delta \equiv \bar{x} - x, \tag{1.12}$$

and

$$y \equiv \nabla_x\mathcal{F}(\bar{x}) - \nabla_x\mathcal{F}(x). \tag{1.13}$$

Then the quasi-Newton condition can be expressed as

$$\bar{B}\delta = y. \tag{1.14}$$

The quasi-Newton condition (1.11, 1.14) comes from the first-order Taylor series approximation to the gradient of the objective function:

$$\nabla \mathcal{F}(\bar{x}) = \nabla \mathcal{F}(x) + \nabla^2 \mathcal{F}(x)(\bar{x} - x) + o(\|\bar{x} - x\|).$$

The error is zero for quadratic functions.

Equation (1.14) does not determine $\bar{B}$ uniquely. One can choose the unique update that minimizes the complexity or size of the update E, for example, while satisfying (1.14). Towards this end, we may prefer the *SR1* update, which is the unique *symmetric, rank-one* update to satisfy (1.14). A quasi-Newton update that can be used to maintain positive definiteness is the *Broyden-Fletcher-Goldfarb-Shanno (BFGS)* update. For this reason, and because it has proven effective in practice for unconstrained and linearly constrained problems, the BFGS update is widely used in SQP methods. BFGS is a rank-two update.

Most quasi-Newton methods, such as BFGS and SR1, are special cases of the one-parameter family of updates introduced by Broyden [Bro70]. The BFGS update is given by

$$\bar{B} = B - \frac{B\delta\delta^T B}{\delta^T B\delta} + \frac{yy^T}{y^T\delta}. \tag{1.15}$$

If B is a positive-definite matrix, it is well known that the BFGS update maintains positive definiteness if and only if $y^T\delta$ is positive. Fortunately, for unconstrained minimization it is possible to terminate the linesearch where $y^T\delta$ is positive. For unconstrained minimization, it can be shown that a Broyden method with exact linesearches terminates after at most n iterations on a quadratic function in n variables with a positive-definite Hessian, and that if the number of iterations reaches n, the Broyden approximation equals the exact Hessian. Note that this result requires exact line searches, and that given a random independent set of n directions such an update is not guaranteed to converge.

For more background on quasi-Newton methods see Gill, Murray and Wright [GMW81], Dennis and Schnabel [DS83], Fletcher [Fle81], and Fletcher[Fle87].

Quasi-Newton extension to constrained optimization

The idea of using a quasi-Newton update in an SQP method to estimate the Hessian of the Lagrangian was proposed by Murray [Mur69]. The most common approach, which we call the *direct approximation*, models the Lagrangian Hessian $\nabla^2_{xx}\mathcal{L}(x, \pi)$ with a single matrix B that combines the Lagrange multiplier estimates with the second-derivative estimates for the objective and constraints. The quasi-Newton condition (1.14) is extended to constrained optimization by redefining the gradient difference as

$$y = \nabla_x \mathcal{L}(\bar{x}, \pi) - \nabla_x \mathcal{L}(x, \pi). \tag{1.16}$$

The BFGS update (1.15) can be applied with this definition of y to update the matrix B.

However, one of the properties that make BFGS methods appealing for unconstrained problems—its maintenance of positive definiteness of B—is no longer assured. Since with nonlinear constraints the Hessian of the Lagrangian is usually positive definite only in a subspace, the linesearch cannot always be terminated where $y^T\delta > 0$. When $y^T\delta \leq 0$, either y must be modified or the update must be skipped. This could happen at every iteration, at the cost of a poor Hessian approximation. For linear constraints, it could also happen that $y^T\delta \leq 0$, but only when a new bound or constraint is encountered in the linesearch, and at a later iteration the update can be performed.

Large-scale quasi-Newton approaches

Problems generated by real-world applications are often large, but sparse in terms of the number of variables appearing in any single constraint or the number of constraints involving any single variable. Methods have been developed to reduce the storage and effort required to handle large-scale problems, including limited-memory methods, reduced Hessian approximations, sparse quasi-Newton approximations, and partial separability.

1.3.4 Solving the QP subproblem

When discussing algorithms to solve the QP subproblem, we convert the nonlinear program to the form BNP having equality constraints and simple bounds. This naturally results in only lower bounds on the QP subproblem variables:

$$\boxed{\text{BQP} \qquad \begin{array}{ll} \underset{p}{\text{minimize}} & g^Tp + \tfrac{1}{2}p^THp \\ \text{subject to} & Ap = -c, \quad p \geq -x. \end{array}} \tag{1.17}$$

When $H \succ 0$ and the minimizer of the QP is used to define the search direction, it is not necessary in a theoretical discussion of an SQP algorithm to define how the QP is solved. If $H \not\succ 0$, there may be more than one local minimizer, so the method used to solve the subproblem, and well as the method to compute an initial starting point for EQP, must be explicitly defined. Moreover, for the purpose of computing a search direction of sufficient descent for the merit function, just having a minimizer may not be enough. More details on finding search directions from QP's are in Chapter 2.

Null-space active set methods

Null-space active-set methods are an efficient way of solving positive-definite QP's when the dimension of the null-space is not too large (≤ 1000) [GMW81]. A typical active-set procedure starts with a feasible point p_0 and a working set (a subset of the active constraints and bounds). Variables whose bounds are in the active set are called *fixed*, and the others are called *free*.

The method proceeds to move to a constrained stationary point of the QP by holding the fixed set of variables constant and temporarily ignoring the other bounds. Consider the equality-constrained QP (EQP) defined by the free variables at the current point (see below). If the solution d^* to EQP is feasible with respect to all the bounds, the full step of length one is taken and a stationary point is reached for the QP. Otherwise the maximum feasible step with respect to the bounds is taken along the search direction d^*, and a bound is added to the working set. This sequence repeats until the full step is taken. At the stationary point, if for any bound there exists a negative Lagrange multiplier, the associated bound is dropped from the working set and the procedure starts over. If all multipliers are positive, the algorithm stops. If the QP is feasible and has no degenerate vertices, this procedure terminates in a finite number of iterations.

The EQP is a problem in the current set of *free* variables:

$$\boxed{\begin{array}{ll} \underset{d \in I\!R^{\bar{n}}}{\text{minimize}} & g_{FR}^T d + \frac{1}{2} d^T H_{FR} d \\ \text{subject to} & A_{FR} d = 0, \end{array}}$$

where d are the free variables from (1.17), A_{FR} is the Jacobian in the free variables, H_{FR} is the Hessian in the free variables, g_{FR} is the gradient of the QP objective computed at the current point in the free variables, and $\bar{n}$ is the number of free variables. The solution to this EQP is given by

$$d^* = Z_{FR} d, \qquad \text{where} \qquad Z_{FR}^T H_{FR} Z_{FR} d = -Z_{FR}^T g_{FR},$$

and Z_{FR} is a basis for the null-space of the rows of A_{FR}. For this process to work, H and H_{FR} do not need to be positive definite, but the *reduced Hessian* $Z_{FR}^T H_{FR} Z_{FR}$ needs to be positive definite on every subspace encountered. Alternatively, d^* may be found by solving the KKT system

$$\begin{pmatrix} H_{FR} & A_{FR}^T \\ A_{FR} & 0 \end{pmatrix} \begin{pmatrix} d^* \\ -\mu^* \end{pmatrix} = - \begin{pmatrix} g_{FR} \\ 0 \end{pmatrix}. \tag{1.18}$$

directly for d^* and the Lagrange multipliers μ^*.

For more details on the active-set method to solve a strictly convex QP, see Chapter 2.

Search directions not based on QP minimizer

In two recent papers, Murray and Prieto describe SQP algorithms where search directions are based not on the minimizer of the QP but on a stationary point of the QP combined with a descent direction formed from negative multiplier estimates [MP95, MP99]. Their SQP algorithms are proven to converge despite using only incomplete solutions of the subproblems.

In the first paper [MP95], Murray and Prieto prove convergence results for an SQP algorithm ETSQP that uses a positive-definite quasi-Newton approximation and permits *early termination* of the active-set method to solve the QP subproblem at any stationary point.

In the second paper [MP99], they describe an SQP algorithm SQP2D that replaces the quasi-Newton approximation with a Hessian based on exact second derivatives. To manage the nonconvex QP subproblems that result, this algorithm actually requires early termination of the subproblems. An adaptation of their algorithm called IDSQP is described in Chapter 2 and is the basis of a method we propose in Chapter 5 to find a search direction from a nonconvex QP.

1.4 Main contributions

1.4.1 Disaggregated Hessian approximations

In Chapters 3 and 4, we explore how best to approximate the Hessian of the Lagrangian, while addressing the issues of asymptotic convergence and sparsity. The direct quasi-Newton approximation has not been shown to converge to the Hessian (and hence does not satisfy conditions for superlinear convergence) even on simple problems, and in certain cases a poor convergence rate does eventuate.

We propose to *disaggregate* the Hessian approximation, that is, to approximate the individual Hessians of $\nabla^2 F$ and $\nabla^2 c_i$ $(i = 1, \ldots, m)$ and form the weighted sum, instead of approximating the Hessian of the Lagrangian directly. This approach is an application of partial separability to quasi-Newton approximation of the Hessian of the Lagrangian, and as such it addresses the challenges presented by sparse problems. As a side-effect, it becomes necessary to relax the requirement for a positive-definite approximation.

There are several advantages to this approach. First, an indefinite Hessian approximation has the potential to be a more accurate estimate of the exact Hessian. If second derivatives of some of the functions are known, they can be used. The approximations of the individual Hessians are independent of the Lagrange multiplier estimates, so the estimate of $\nabla^2 \mathcal{L}$ can respond quickly to changes in the multiplier estimates. Lastly, if the individual Hessians are quadratic functions, then after a small number of iterations the matrices will be essentially exact.

1.4.2 Nonconvex QP subproblems

H might not be positive definite (or even semidefinite) if it is based on exact derivatives, finite-differences, or a quasi-Newton method that does not ensure positive definiteness. In Chapter 5, we describe how to find a search direction from a nonconvex QP.

Our approach is to modify the problem, but only as needed, during the null-space active-set method. It is desirable not to modify the Hessian in the neighborhood of the solution, where the reduced Hessian is expected to be positive definite. Away from the solution, if the QP is modified to make the reduced Hessian positive definite at the initial point, it can be shown (see [MP99]) that if a search direction is defined based on the first constrained stationary point, the method does converge. We describe how to continue past the first stationary point.

1.5 Overview of remaining chapters

- In Chapter 2 we describe two "framework" algorithms: a basic SQP algorithm for positive-definite Hessian approximations and Murray and Prieto's SQP algorithm that allows indefinite Hessian approximations [MP99] but terminates QP subproblems at the first stationary point.

- In Chapter 3 we describe the direct approximation for the Hessian of the Lagrangian.
- In Chapter 4 we present the disaggregated approximation for the Hessian of the Lagrangian.
- In Chapter 5 we discuss the issues introduced by the "nonconvex" subproblem and present a new algorithm to find a direction of sufficient descent for the merit function from a nonconvex QP subproblem that extends the QP subproblem past the first stationary point.
- Chapter 6 describes a MATLAB implementation of our algorithm and presents some numerical results. We summarize our findings.

Chapter 2

SQP Framework Algorithms

In this chapter, we outline two framework SQP algorithms. PDSQP requires the Hessian or Hessian approximation to be positive definite at every iteration, and consequently unique solutions to the QP subproblems are guaranteed. IDSQP does not require the Hessian or Hessian approximation to be positive definite at every iteration, so a unique (or even bounded) solution to the original QP subproblem is no longer guaranteed.

IDSQP is based on work by Murray and Prieto [MP95, MP99], who generalized their early-termination SQP algorithm ETSQP to incorporate exact second derivatives. The second-derivative algorithm, called SQP2D, combines a descent direction with a direction of negative curvature and performs a curvilinear search on the merit function at each major iteration. What interests us, in the context of this thesis, is that since the exact Hessians are not necessarily positive definite, Murray and Prieto developed a new strategy for finding a descent direction from a *nonconvex* QP subproblem. We present a simpler version of their second-derivative algorithm SQP2D that uses indefinite quasi-Newton approximations instead of exact second derivatives. We eliminate the use of directions of negative curvature from SQP2D and replace the curvilinear search with a standard linesearch. We call the remaining algorithm IDSQP, for "indefinite" or "indeterminate" SQP.

2.1 Positive-definite SQP algorithm (PDSQP)

PDSQP requires the Hessian or Hessian approximation to be positive definite at every iteration, and consequently unique solutions to the QP subproblems are guaranteed. How the

QP is solved is irrelevant to the convergence property of the SQP algorithm, but it is generally an iterative process with minor iterations (i.e., "QP iterations"). The quasi-Newton approximation to the Hessian of the Lagrangian is usually maintained using the BFGS quasi-Newton update to produce a positive-definite matrix B_k; this process is described in more detail in Chapter 3.

The main steps of each major iteration of algorithm PDSQP are:

- Solve the QP subproblem for a search direction p_k and multiplier estimate μ_k.
- Compute optimal slacks and slack search direction q_k.
- Compute multiplier search direction ξ_k.
- Update penalty parameter ρ_k.
- Select steplength α_k using a linesearch on the merit function.
- Update iteration values x_{k+1}, g_{k+1}, c_{k+1}, A_{k+1}, λ_{k+1}.
- Update Hessian approximation B_{k+1}.

2.1.1 Merit function and linesearch

What mainly distinguishes one positive-definite SQP algorithm from another is the choice of merit function used in the linesearch, which determines the steplength.

The SQP algorithms we consider in detail feature the *smooth augmented Lagrangian merit function*

$$L_A(x, \lambda, s, \rho) = f(x) - \lambda^T(c(x) - s) + \frac{1}{2}\rho(c(x) - s)^T(c(x) - s), \qquad (2.1)$$

where $\rho > 0$ is the penalty parameter. The search space for this merit function includes the iterate x and, in addition, the Lagrange multiplier estimates λ and the slack variables $s \geq 0$. This merit function was suggested by Gill *et al.* [GMSW92] and a version of it has been successfully implemented in the dense SQP code NPSOL [GMSW86] and the large-scale SQP code SNOPT [GMS97]. A discussion of the smooth augmented Lagrangian function and other merit functions can be found in [Mur97].

Some notation

Components of the search direction on the subspaces corresponding to x, λ, and s will be denoted p, ξ, and q. For a fixed penalty parameter ρ, the value of the merit function ϕ as a function of the steplength α is denoted

$$\phi(\alpha; x, p, \lambda, \xi, s, q, \rho) \equiv L_A(x + \alpha p, \lambda + \alpha\xi, s + \alpha q, \rho). \tag{2.2}$$

We may abbreviate $\phi(\alpha; x_k, p_k, \lambda_k, \xi_k, s_k, q_k, \rho_k)$ by $\phi_k(\alpha)$ or by $\phi(\alpha)$ when the meaning is clear.

Slack variables

The slack variables appear within separable quadratic functions in the merit function; hence, their optimal values can be computed. At a given iterate, the optimal values of the slacks that minimize the merit function are

$$s = \begin{cases} \max(0, c) & \text{if } \rho = 0, \\ \max(0, c - \lambda/\rho) & \text{otherwise.} \end{cases} \tag{2.3}$$

Search direction

The search direction p comes directly from the QP subproblem. At each major iteration, the search direction q in the space of slack variables is computed to satisfy

$$Ap - q = -(c - s),$$

which ensures that the slacks remain at their optimal values for any linear constraints. The search direction in the space of the multiplier estimates is $\xi = \mu - \lambda$.

Derivative of the merit function

Consider the gradient of the merit function L_A with respect to x, λ, and s,

$$\nabla L_A(x, \lambda, s) = \begin{pmatrix} g(x) - \nabla c(x)^T\lambda + \rho\nabla c(x)^T(c(x) - s) \\ -(c(x) - s) \\ \lambda - \rho(c(x) - s) \end{pmatrix}. \tag{2.4}$$

We may denote the derivative of ϕ_k with respect to α by ϕ_k'. It follows from (2.1) and (2.2) that $\phi'(0)$ is given by

$$\begin{aligned}\phi'(0) &= g^Tp - p^TA^T\lambda + \rho p^TA^T(c-s) - (c-s)^T\xi + \lambda^Tq - \rho q^T(c-s)\\ &= g^Tp + (2\lambda - \mu)^T(c-s) - \rho\|c-s\|^2,\end{aligned}$$

where g, $A = \nabla c(x)$, and c are evaluated at x.

Penalty parameter

The penalty parameter is adjusted as necessary at each iteration to ensure that (p, ξ, q) satisfies the desired descent condition

$$\phi'(0) \leq -\frac{1}{2}p^TBp. \tag{2.5}$$

If $\|c-s\| = 0$, it follows from the definition of the slack variables that $p_0 = 0$ is an initial feasible point for the QP subproblem. Let the QP objective function be denoted $\psi(p) \equiv p^Tg + \frac{1}{2}p^THp$. At the solution of the QP, $\psi(p) = p^Tg + \frac{1}{2}p^THp \leq \psi(p_0) = 0$. Hence,

$$\phi'(0) = p^Tg \leq -\frac{1}{2}p^THp,$$

implying that ρ does not need to be modified [MP95, pp. 603].

If $\|c-s\| > 0$, we obtain from (2.5) that for

$$\rho \geq \frac{(g^Tp + \frac{1}{2}p^TBp) + (2\lambda-\mu)^T(c-s)}{\|c-s\|^2}, \tag{2.6}$$

we have

$$\phi'(0) = g^Tp + (2\lambda-\mu)^T(c-s) - \rho\|c-s\|^2 \leq -\frac{1}{2}p^THp,$$

which implies the desired descent condition is satisfied.

In addition to satisfying (2.6), the penalty parameter must be adjusted in a manner that ensures it is not modified too often and maintains a minimum value of $\beta_\rho > 0$.

Steplength

The procedure to compute the steplength $\alpha_k > 0$ uses a linesearch to reduce $\phi_k(\alpha)$, then backtracks to ensure the constraint violation is bounded. The linesearch depends on constants ν and η, and the boundedness of constraint violations depends on β_c, where

$$0 < \nu < \eta < 1 \quad \text{and} \quad \beta_c \geq \|c^-(x_0)\|_\infty.$$

Pseudocode for the linesearch procedure is in Figure 2.1.

Linesearch procedure

if $\phi_k(1) - \phi_k(0) \leq \nu\phi_k'(0)$
  $\hat{\alpha} \leftarrow 1$
else
  Select $\hat{\alpha} \in (0,1)$ to satisfy
    $\phi_k(\hat{\alpha}) \leq \phi_k(0) + \nu\hat{\alpha}\phi_k'(0), \quad |\phi_k'(\hat{\alpha})| \leq -\eta\phi_k'(0)$
end if
while $c(x_k + \hat{\alpha}p_k) \leq -\beta_c e$ **or** $\phi_k(\hat{\alpha}) > \phi_k(0) + \nu\hat{\alpha}\phi_k'(0)$ **do**
  $\hat{\alpha} \leftarrow \hat{\alpha}/2$
end do
$\alpha_k \leftarrow \hat{\alpha}$

Figure 2.1: Linesearch procedure.

2.1.2 Statement of the algorithm

The pseudocode for algorithm PDSQP is in Figure 2.2. We assume values are given for x_0, g_0, A_0, c_0, an initial multiplier estimate $\lambda_0 \geq 0$, a positive-definite matrix B_0, an initial penalty parameter $\rho_{-1} \geq 0$, and a multiplier estimate bound $\beta_\mu \geq \|\lambda_0\|$.

2.1.3 Solving the QP subproblem

Although the method of solving the positive-definite QP subproblem is unimportant because the solution is unique, for reference we provide an active-set procedure PDQP for solving a positive-definite QP. The procedure PDQP is presented in Figure 2.3.

As in Chapter 1, we consider the QP subproblem to be in the form BQP (1.17) introduced

Algorithm PDSQP

$k \leftarrow 0$
repeat
  $H \leftarrow B_k$
  Obtain the search direction p_k by calling PDQP to solve the QP subproblem

$$\begin{array}{ll} \min_p & g_k^T p + \frac{1}{2} p^T H p \\ \text{s.t.} & A_k p + c_k \geq 0 \end{array}$$

  Optimize the slack variables s_k (cf. (2.3))
  Form search direction for slacks $q_k \leftarrow A_k p_k + c_k - s_k$
  Form μ_k, an estimate of λ^* such that $\|\mu_k\| \leq \beta_\mu$
  Form multiplier search direction $\xi_k \leftarrow \mu_k - \lambda_k$
  if $\phi_k'(0) \leq -\frac{1}{2} p_k^T B_k p_k$
    $\rho_k \leftarrow \rho_{k-1}$
  else
    Increase ρ_k by at least a factor of 2, and
    large enough to satisfy (2.5) and (2.6)
  end
  Perform linesearch on the merit function to choose α_k (Figure 2.1).

$$\begin{pmatrix} x_{k+1} \\ \lambda_{k+1} \end{pmatrix} \leftarrow \begin{pmatrix} x_k \\ \lambda_k \end{pmatrix} + \alpha_k \begin{pmatrix} p_k \\ \xi_k \end{pmatrix}$$

  Evaluate g_{k+1}, c_{k+1}, and A_{k+1}
  Update B_k to form B_{k+1}
  $k \leftarrow k + 1$
until *convergence*

Figure 2.2: Algorithm PDSQP.

in Section 1.3.4:

BQP	$\underset{p}{\text{minimize}} \quad \psi(p) \equiv g^T p + \frac{1}{2} p^T H p$ subject to $\; Ap = -c, \quad p \geq -x.$

Notation

To avoid additional notation, there is an overlap of notation between the descriptions of the SQP algorithm and the QP algorithm, with p, A, and Z having separate meanings in the context of the major (SQP) iterations and the minor (QP) iterations. To help avoid confusion, the subscript k is used only to denote major iterations, while the subscript j is used only in the QP iterations. Also note that in the minor iterations, the QP Hessian is H, while the Hessian approximation in the major iterations is denoted B.

The following notation and definitions will be used whenever we solve or compute a search direction from BQP. Suppose an initial feasible step p_0 is given. Let A_0 be the matrix composed of the columns of A whose variables are free at p_0, and let Z_0 be the matrix whose columns are a basis for the null space of A_0. Extending our notation, let A_j denote the matrix composed of the columns of A whose variables are free at the jth iteration and let Z_j be the matrix whose columns span the null space of A_j. Note that we define "iteration" such that j is incremented whenever there is a change to the active set. This includes both when a step is taken and p_j changes, as well as when a constraint is deleted from the active set but p_j does not change.

We introduce a permutation matrix

$$P_j \equiv \begin{pmatrix} \bar{P}_j & \widehat{P}_j \end{pmatrix} \tag{2.7}$$

such that the elements of $v \equiv P_j^T u$, where $u_i = i$, give the reordering of the variables by placing the indices of the free variables first followed by those of the fixed variables.

Define H_j to be the Hessian restricted to the free variables, $H_j \equiv \bar{P}_j^T H \bar{P}_j$. Let $g_j = \bar{P}_j^T g$. Note that p_j is the jth iterate in the full variable space.

Algorithm PDQP for Positive-Definite Quadratic Programming

Obtain an initial feasible point p_0
Identify initial working set and compute P_0, A_0, Z_0, and H_0
$j \leftarrow 0$
repeat
  repeat
    $g_j \leftarrow \bar{P}_j^T(g + Hp_j)$
    stationary_point $\leftarrow Z_j^T g_j = 0$
    if not *stationary_point* **then**
      Solve for d satisfying $\begin{pmatrix} H_j & A_j^T \\ A_j & 0 \end{pmatrix} \begin{pmatrix} d \\ -\pi \end{pmatrix} = \begin{pmatrix} -g_j \\ 0 \end{pmatrix}$
      $\bar{d} \leftarrow P_j^T \begin{pmatrix} d \\ 0 \end{pmatrix}$
      $\gamma_M \leftarrow \min_\tau \left\{ \dfrac{x_\tau + p_\tau}{|\bar{d}_\tau|} \mid \bar{d}_\tau < 0 \right\}$
      hit_constraint $\leftarrow \gamma_M < 1$
      $\gamma \leftarrow$ **if** *hit_constraint* **then** γ_M **else** 1
      $p_{j+1} \leftarrow p_j + \gamma \bar{d}$
      Update working set and compute P_{j+1}, A_{j+1}, Z_{j+1}, and H_{j+1}
      $j \leftarrow j + 1$
    end
  until *stationary_point*
  Compute the Lagrange multipliers $\tilde{\sigma}_{FX}$ for the active bounds from (2.11)
  $\tilde{\sigma}_\tau \leftarrow \min_i \tilde{\sigma}_i$
  converged $\leftarrow \tilde{\sigma}_\tau \geq 0$
  if not *converged* **then**
    Delete bound with multiplier $\tilde{\sigma}_\tau$
    Update working set and compute P_{j+1}, A_{j+1}, Z_{j+1}, and H_{j+1}
    $j \leftarrow j + 1$
  end
until *converged*

Figure 2.3: Algorithm PDQP.

Constrained stationary point

Moving to a constrained stationary point is a key component of algorithm PDQP. To explain this key component in slightly more general terms, we consider the Hessian of the BQP to be $H^{(0)}$ instead of H, where $H^{(0)}$ may be indefinite as long as any reduced Hessian encountered is positive definite. As before, let $H_j^{(0)}$ be the Hessian restricted to the free variables at iteration j.

The necessary conditions that hold at a constrained stationary point $\tilde{p}$ of BQP (1.17) are, for some multiplier vectors π and $\tilde{\sigma}$:

$$\begin{aligned} & g + H^{(0)}\tilde{p} = A^T\pi + \tilde{\sigma} \\ & A\tilde{p} + c = 0 \\ & \tilde{p} \geq -x \\ & \tilde{\sigma}^T(\tilde{p} + x) = 0. \end{aligned} \tag{2.8}$$

If at the stationary point we partition the variables into *fixed (FX)* and *free (FR)* using the permutation matrix P from (2.7), the Hessian can be written

$$\begin{aligned} H^{(0)} &= \begin{pmatrix} \bar{P}^T H^{(0)} \bar{P} & \bar{P}^T H^{(0)} \widehat{P} \\ \widehat{P}^T H^{(0)} \bar{P} & \widehat{P}^T H^{(0)} \widehat{P} \end{pmatrix} \\ &\equiv \begin{pmatrix} H_{FR}^{(0)} & H_{FRX}^{(0)T} \\ H_{FRX}^{(0)} & H_{FX}^{(0)} \end{pmatrix}, \end{aligned} \tag{2.9}$$

and the submatrices H_{FX}, H_{FR}, and H_{FRX} are defined accordingly.

In terms of the free and fixed variables, the constrained stationary point $(\tilde{p}_{FR}, \tilde{p}_{FX})$, π, and $\tilde{\sigma}$ then satisfy

$$\begin{pmatrix} A_{FR} & A_{FX} \\ 0 & I_{FX} \end{pmatrix} \begin{pmatrix} \tilde{p}_{FR} \\ \tilde{p}_{FX} \end{pmatrix} = \begin{pmatrix} -c \\ -x_{FX} \end{pmatrix} \tag{2.10}$$

$$\begin{pmatrix} A_{FR}^T & \\ A_{FX}^T & I_{FX} \end{pmatrix} \begin{pmatrix} \pi \\ \tilde{\sigma} \end{pmatrix} = \begin{pmatrix} g_{FR} \\ g_{FX} \end{pmatrix} + \begin{pmatrix} H_{FR}^{(0)}\tilde{p}_{FR} - H_{FRX}^{(0)T} x_{FX} \\ H_{FRX}^{(0)}\tilde{p}_{FR} - H_{FX}^{(0)} x_{FX} \end{pmatrix}.$$

Step to the stationary point

The null-space active-set method computes the step to the stationary point by starting from an initial feasible point and then forming at every iteration an equality-constrained QP (EQP) in the current set of free variables. The solution to the EQP is used as a search direction for the active-set method. Suppose there are $\bar{n}$ free variables. At iteration j, the EQP (where d represents the free variables only) is

$$\text{EQP} \qquad \begin{array}{ll} \underset{d \in I\!R^{\bar{n}}}{\text{minimize}} & g_j^T d + \frac{1}{2} d^T H_j^{(0)} d \\ \text{subject to} & A_j d = 0, \end{array}$$

where $g_j = \bar{P}_j^T (g + H^{(0)} p_j)$.

For a minimizer d^* of the EQP to exist, $Z_j^T H_j^{(0)} Z_j$ must be positive definite (but $H_j^{(0)}$ is not necessarily positive definite). The KKT system representing the necessary conditions for the minimizer of the EQP are

$$\begin{pmatrix} H_j^{(0)} & A_j^T \\ A_j & 0 \end{pmatrix} \begin{pmatrix} d^* \\ -\pi \end{pmatrix} = \begin{pmatrix} -g_j \\ 0 \end{pmatrix}.$$

The solution to the EQP can also be represented in terms of the null-space matrix Z_j, as

$$d^* = Z_j d^Z, \qquad \text{where} \qquad Z_j^T H_j^{(0)} Z_j d^Z = -Z_j^T g_j.$$

After the direction d^* is computed, the iterate p is incremented by a step taken, in that direction, of length $\gamma \le 1$

$$p \leftarrow p + \gamma P^T \begin{pmatrix} d^* \\ 0 \end{pmatrix}.$$

The steplength γ is chosen to maintain feasibility with respect to all bounds. Eventually either a vertex is reached or the unit step is taken, at which point the requirements for a stationary point (2.10) are satisfied at the current point $\tilde{p} = p$ and π.

Releasing a variable from its bound

The multipliers $\tilde{\sigma}_{FX}$ for the active bounds are computed from

$$\tilde{\sigma}_{FX} = g_{FX} + H^{(0)}_{FRX}\tilde{p}_{FR} - H^{(0)}_{FX}x_{FX} - A^T_{FX}\pi, \tag{2.11}$$

or in the extended multipliers,

$$\tilde{\sigma} = g + H^{(0)}(\bar{P}_j\bar{P}^T_j)\tilde{p} - H^{(0)}(\widehat{P}_j\widehat{P}^T_j)x - A^T\pi. \tag{2.12}$$

If one of the multipliers is negative, j is incremented, the corresponding variable is released from its bound, and new values are computed for P_j, A_j, Z_j, and H_j. The algorithm terminates with the solution when $\tilde{\sigma}_{FX} \geq 0$.

2.1.4 Convergence results for positive-definite SQP algorithm

The convergence results we give for PDSQP are based on results by Murray and Prieto [MP95] for a related but more general algorithm. What makes their algorithm (called ETSQP) more general is that it allows a wider variety of multiplier estimates and allows an incomplete solution of the subproblem. ETSQP allows any constrained stationary point of the QP subproblem, when combined with a descent direction taken from the stationary point and formed using the negative multiplier estimates at the stationary point, to be a search direction for the QP subproblem. Their approach limits the amount of work required in the subproblems by allowing "early termination" (hence the name ETSQP).

Under certain assumptions, global convergence for PDSQP follows from Murray and Prieto's results for ETSQP. In particular, the sequence $\{x_k\}$ generated by PDSQP converges to a unique KKT point and λ_k converges to λ^*. Moreover, under additional assumptions on the quality of the Hessian approximation, it can be shown that the penalty parameter is bounded and the rate of convergence is superlinear.

We list below the precise assumptions needed to prove these results. Assumption A5, that every subproblem has a feasible solution, may be satisfied by modifying the SQP algorithm to allow infeasible constraints (see [MP95, GMS97, Bom99]). With this caveat, the first set of general assumptions all relate to properties of the problem, not the iterates or the algorithm. Assumptions MC3 and HC3 are needed only to prove the superlinear rate of convergence, not for global convergence. Note, it is not necessary to assume the iterates lie in a compact region. See [MP95, MP99] for a more detailed discussion of these assumptions.

Except for assumption HC2 (that the Hessian approximation is positive definite), we shall continue to make these assumptions for all SQP algorithms discussed.

General assumptions

A1. For some constant $\beta_c > 0$, the global minimum of the problem

$$\begin{array}{ll} \underset{x \in I\!R^n}{\text{minimize}} & F(x) \\ \text{s.t.} & c(x) \geq -\beta_c e, \end{array}$$

exists.

A2. There exist no KKT points at infinity for problem NP (we use the definition of "KKT point at infinity" given in [MP95]).

A3. $F(x)$, $c(x)$ and their first and second derivatives are continuous and uniformly bounded in norm on a compact set.

A4. The Jacobian corresponding to the active constraints at all KKT points has full rank.

A5. A feasible point p_{0_k} exists to all the QP subproblems, satisfying

$$\|p_{0_k}\| \leq \beta_{p0}\|\tilde{c}_k^-\|, \qquad g_k^T p_{0_k} \leq \beta_{p0}\|\tilde{c}_k^-\|,$$

for some constant $\beta_{p0} > 0$, where $\tilde{c}_k$ denotes the normalized constraints, $(\tilde{c}_k)_i \equiv c_k/(1 + \|(a_k)_i\|)$, and $(a_k)_i$ is the ith row of A_k.

A6. Strict complementarity holds at all stationary points of NP, including stationary points at infinity, if they exist.

A7. The reduced Hessian of the Lagrangian is nonsingular at all KKT points.

Assumptions on the multiplier estimates

MC1. The estimates μ_k are uniformly bounded in norm; that is, $\|\mu_k\| \leq \beta_\mu < \infty$.

MC2. The complementarity condition $\mu_k^T(A_k p_k + c_k) = 0$ is satisfied at all major iterations.

Assumptions on the Hessian approximation

HC1. $\beta_{svB} < \infty$ is an upper bound on the largest eigenvalue of $\{B_k\}$.

HC2. $\beta_{lvB} > 0$ is a lower bound on the smallest eigenvalue of $\{B_k\}$.

Assumptions needed to prove superlinear rate of convergence

MC3. $\|\mu_k - \lambda^*\| = O(\|x_k - x^*\|)$, where λ^* denotes the multiplier vector associated with a KKT point x^* closest to x_k.

HC3. Following Boggs, Tolle and Wang [BTW82], we assume

$$\|\widehat{Z}_j^T(B_k - \nabla_x^2\mathcal{L}(x_k, \lambda))p_k\| = o(\|p_k\|),$$

where B_k is the approximation to $\nabla_x^2\mathcal{L}(x_k, \lambda)$, and $\widehat{Z}_k$ is a basis for the null space of $\widehat{A}_k$, the Jacobian at x_k of those constraints active at x^*, that is bounded in norm and has its smallest singular value bounded away from 0.

2.2 SQP algorithm with indefinite Hessian approximations

The *indefinite* SQP algorithm does not require the Hessian or Hessian approximation to be positive definite at every iteration, so a bounded *solution* to the original QP subproblem is no longer guaranteed, let alone a unique solution. Consequently, it does matter both how the QP is solved and how the initial feasible point is computed. Depending on convenience and availability, the QP Hessian may be formed from exact second derivatives or approximated by any general quasi-Newton update (for more detail refer to Chapter 4).

The indefinite and positive-definite algorithms (IDSQP and PDSQP) are similar in many ways. They both use the same merit function (2.1). The main differences have to do with the way of computing the search direction from the subproblem and the algorithm for adjusting the penalty parameter ρ to ensure a descent direction at every iteration.

As before, we consider the QP to have the form BQP (1.17), which was introduced in Section 1.3.4 and repeated in Section 2.1.3.

2.2.1 Computing a search direction from the indefinite QP subproblem

Murray and Prieto's strategy for computing a search direction from a nonconvex QP is based on modifying the QP Hessian at the initial feasible point so the initial reduced Hessian is positive definite. For the modified QP subproblem there exists a constrained stationary point, which can be found by the null-space active-set method. Combining the step to the stationary point with a certain direction of descent provides a satisfactory search direction for the merit function.

The main steps of the procedure IDQP to compute a search direction are:

- Find an initial feasible step p_0 for the BQP.
- Modify the initial reduced Hessian to be positive definite.
- Compute the step to the (first) stationary point.
- Compute a descent step from the stationary point.

Initial feasible point

The initial point p_0 must be feasible ($Ap_0 = -c$ and $p_0 \geq -x$) and for some $\beta_{p0} > 0$ satisfy

$$\|p_0\| \leq \beta_{p0}(\|\tilde{c}\|^2 + \|x^-\|^2)^{1/2}, \qquad g^T p_0 \leq \beta_{p0}(\|\tilde{c}\|^2 + \|x^-\|^2)^{1/2}. \tag{2.13}$$

When the minimizer of a positive-definite QP is used as the search direction, then since the solution p is unique, the choice of p_0 is irrelevant. If we determine the search direction from a stationary point that is not a minimizer, the sequence of stationary points that we compute depends directly on the value of p_0. These conditions on p_0 ensure that all stationary points are satisfactory points at which to terminate the solution process.

Modified reduced Hessian

If the initial reduced Hessian $Z_0^T H Z_0$ is not positive definite, H is modified. Let $H^{(0)}$ be a modification of H for which $Z_0^T H_0^{(0)} Z_0 \succ 0$. If $Z_0^T H Z_0$ can be formed, Murray and Prieto suggest determining $H^{(0)}$ using the method described in [FGM95]. Let $\psi^{(0)}$ be the corresponding QP objective

$$\psi^{(0)}(p) = \frac{1}{2}p^T H^{(0)} p + g^T p.$$

In the large-scale case, the issue is more difficult, since merely forming $Z_0^T H Z_0$ may be prohibitively expensive. Fortunately, the nature of this matrix may be deduced from a factorization of the KKT matrix

$$\begin{pmatrix} H & A_0^T \\ A_0 & 0 \end{pmatrix}.$$

Forsgren and Murray [FM93] describe how to determine $H^{(0)}$ using the KKT matrix.

Alternatives for how to compute $H^{(0)}$ in both the large-scale and the small-scale cases are described in more detail in Chapter 5.

Existence of constrained stationary point

We shall prove a constrained stationary point exists in the case where $H^{(0)}$ is indefinite.

Lemma 2.2.1 *Consider the problem BQP with objective $\psi^{(0)}(p)$. If $Z_0^T H^{(0)} Z_0$ is positive definite, the active-set method starting from p_0 will reach a constrained stationary point (2.8) in a finite number of iterations.*

Proof. The initial working set, the set of free and fixed variables, and the nullspace matrix Z_0 all depend on the initial feasible point p_0. A consequence of $H^{(0)}$ being chosen such that $Z_0^T H_0^{(0)} Z_0 \succ 0$ is that the solution to the EQP on the initial fixed variables,

$$\boxed{\begin{array}{ll} \underset{p \in \mathbb{R}^n}{\text{minimize}} & g^T p + \frac{1}{2} p^T H^{(0)} p \\ \text{subject to} & Ap = 0, \quad p_{FX} = -x_{FX}, \end{array}}$$

is bounded below, hence a minimizer exists to this problem. If the minimizer satisfies $p_{FR} \geq -x_{FR}$, a constrained stationary point has been reached. Otherwise, a step towards the minimizer is taken, the working set changes and the problem is solved again.

From p_0, bounds are not deleted from the working set until a minimizer is reached. Since the first EQP is bounded below, when additional bounds are added the EQP remains bounded, and the steps taken are in a subset of the initial free variables. This manner of changing the working set ensures the reduced Hessian is positive definite on every subspace encountered. The algorithm continues until the working set defines a unique point or the unit step is taken so a minimizer is reached. In either case we have reached a constrained stationary point in a finite number of iterations. ∎

Acceptable stationary stopping points

Under a certain stronger positivity assumption on the Hessian of the modified QP, the proof of convergence for Murray and Prieto's second-derivative algorithm SQP2D allows the algorithm to continue to a stationary point $\tilde{p}$ past the first one. In such a case, the algorithm would proceed as in PDQP, where the bound corresponding to the minimum eigenvalue is deleted from the set of active bounds, the reduced Hessian is updated, etc., until the next stationary point is reached. We need to ensure that the reduced Hessian is positive definite on all subspaces encountered, otherwise a minimizer on the subspace would not exist.

Suppose the Hessian has been modified at most once, at the initial feasible point. Let $\tilde{p}$ be a stationary point, and let $\tilde{H} \equiv H^{(0)}$ be the Hessian at $\tilde{p}$.

The general *positivity assumption* made in [MP99] requires that the modified Hessian be sufficiently positive definite on the union of the nullspaces encountered when determining a stationary point. In other words, $V^T\tilde{H}V \succ 0$, where the columns of V are a basis for all null-spaces encountered. For example, this ensures that $(\tilde{p}-p_0)^T\tilde{H}(\tilde{p}-p_0) \succ 0$, where p_0 is the initial feasible point and $\tilde{p}$ is the final stationary point.

Descent step off stationary point

In SQP2D, when the stationary point $\tilde{p}$ is not a minimizer, it is combined with a multiple $\hat{\gamma}$ of a sufficient descent step u that moves off several of the constraints with negative multipliers.

In the discussion that follows, let $\tilde{p}$ be a stationary point but not a minimizer of the QP. Let $\tilde{\sigma}$ be the multiplier estimates at $\tilde{p}$ for the bounds and let $\tilde{\pi}$ be the multiplier estimates at $\tilde{p}$ for the general constraints.

Let $\tilde{H}$ be the Hessian at $\tilde{p}$ (for algorithm IDQP, $\tilde{H} = H^{(0)}$, but in Chapter 5 it could vary). Let u_{FX} be the components of u corresponding to fixed variables at $\tilde{p}$, and u_{FR} be the components of u corresponding to free variables. The descent step u is required to be a direction of sufficient descent from $\tilde{p}$ satisfying the following conditions:

DD1. The direction is feasible and of unit length:
$Au = 0$, $u_{FX} \geq 0$, and $\|u\|_\infty = 1$.

DD2. The rate of descent along u is "sufficiently" large, i.e., for some $0 < \beta_u \leq 1$, u satisfies

$$\tilde{g}^T u \leq \beta_u \tilde{g}^T u^*, \tag{2.14}$$

where $\tilde{g} \equiv \tilde{H}\tilde{p} + g$ and u^* solves

$$\begin{array}{ll} \min_u & \tilde{g}^T u \\ \text{s.t.} & Au = 0 \\ & u_{FX} \geq 0 \\ & \|u\|_\infty \leq 1. \end{array} \tag{2.15}$$

Note that (2.14) and (2.15) imply $\tilde{g}^T u \leq 0$.

Murray-Prieto descent direction

The following descent direction was suggested in [MP99]. Assuming A has a bounded condition number, let $(\hat{u}, \hat{\pi}, \hat{\sigma})$ be the solution of

$$\begin{aligned} \tilde{H}\hat{u} - A^T\hat{\pi} - \hat{\sigma} &= -\tilde{g} \\ A\hat{u} &= 0 \\ \hat{u}_{FX} &= \tilde{\sigma}^-_{FX}. \end{aligned} \tag{2.16}$$

Since $\tilde{p}$ is not a minimizer of the QP, $\min_i \tilde{\sigma}_i < 0$ (so $\tilde{\sigma}^-_{FX} \neq 0$). System (2.16) is the set of first-order necessary conditions for the EQP

$$\boxed{\begin{array}{ll} \underset{u}{\text{minimize}} & \frac{1}{2}u^T\tilde{H}u + \tilde{g}^T u \\ \text{subject to} & Au = 0, \quad u_{FX} = \tilde{\sigma}^-_{FX}. \end{array}}$$

The following argument shows that the direction $u \equiv \hat{u}/\|\hat{u}\|_\infty$ satisfies conditions **DD1** and **DD2**. First, the direction u is well-defined because under the assumptions on A, $\|\hat{u}\|_\infty$ is bounded. Clearly $\|u\|_\infty = 1$, and the rest of **DD1** follows from (2.16). Since at the stationary point $\tilde{g} = A^T\tilde{\pi} + \tilde{\sigma}$,

$$\tilde{g}^T u = (\tilde{\pi}^T A + \tilde{\sigma}^T)u = \tilde{\sigma}^T u = -\frac{\tilde{\sigma}^T\tilde{\sigma}}{\|\hat{u}\|_\infty} \leq -\frac{(\min_i \tilde{\sigma}_i)^2}{\|\hat{u}\|_\infty}. \tag{2.17}$$

Together with

$$\tilde{g}^T u^* = (\tilde{\pi}^T A + \tilde{\sigma}^T)u^* = \tilde{\sigma}^T u^* = \tilde{\sigma}^T_{FX} u^*_{FX} \geq n \min_i (\tilde{\sigma}_{FX})_i, \tag{2.18}$$

this implies

$$\tilde{g}^T u \leq -\frac{(\min_i \tilde{\sigma}_i)^2}{\|\hat{u}\|_\infty} \leq \left(\frac{\min_i \tilde{\sigma}_i}{n\|\hat{u}\|_\infty}\right) \tilde{g}^T u^*. \tag{2.19}$$

If not for the general constraints, the resulting direction $\hat{u}$ would be precisely a steepest descent direction. One might ask why we select a descent direction that fixes one or more newly freed variables at their multipliers (see (2.16)), and in doing so select the steepest-descent direction in those variables, rather than compute the Newton step to the minimizer. The reason is because this ensures the updated reduced Hessian is still positive definite. In Section 5.4.4, we describe an alternative descent direction that is the Newton step to a minimizer of a problem with a modified Hessian. There we address the problem of the updated reduced Hessian becoming indefinite.

Steplength

After normalizing the descent step $\hat{u}$ so $u \equiv \hat{u}/\|\hat{u}\|$, a steplength $\tilde{\gamma}$ is computed, such that

- $\tilde{p} + \tilde{\gamma} u$ is a feasible point for the QP,
- if $u^T \tilde{H} u > 0$,

  $$\tilde{\gamma} < -\frac{(g + \tilde{H}\tilde{p})^T u}{u^T \tilde{H} u},$$

  that is, the steplength is less than

  $$\arg\min_{\gamma} g^T(\tilde{p} + \gamma u) + \frac{1}{2}(\tilde{p} + \gamma u)^T \tilde{H} (\tilde{p} + \gamma u).$$

- $\tilde{\gamma} < \gamma_M$ ($\tilde{\gamma}$ is bounded above), and
- $\|\tilde{p} + \tilde{\gamma} u\| \geq \|\tilde{p}\|$ (to ensure the resulting direction is sufficiently large).

Summary

At each iteration of **IDSQP** an inner iteration is performed to compute the search direction from the indefinite QP subproblem. The algorithm endeavors to solve the QP subproblem using an active-set method but terminates early in order to satisfy the positivity condition. In Figures 2.4–2.6, we provide pseudocode summarizing the algorithm. It is broken into three parts:

- Main subroutine IDQP: Compute-search-direction-from-indefinite-QP

- Subroutine Move-to-stationary-point

- Subroutine Compute-descent-direction

We assume that positive constants β_{p0}, β_{de}, β_{HZ}, β_{B} and $\gamma_M > 1$ have been defined. The subscript j refers to the QP (inner) iterations.

Algorithm IDQP: Compute-search-direction-from-indefinite-QP

Obtain a feasible point p_0 satisfying (2.13)
Identify initial working set and compute P_0, A_0, Z_0, and H_0
if $\lambda_{\min}(Z_0^T H_0 Z_0) > \beta_{HZ}$, **then**
  $H_0^{(0)} \leftarrow H_0$
else select $H_0^{(0)}$ so that $\lambda_{\min}(Z_0^T H_0^{(0)} Z_0) > \beta_{HZ}$ and $\|H^{(0)}\| \leq \beta_H$
end
$j \leftarrow 0$

$\boxed{\text{Move-to-stationary-point}}$

$\tilde{p} \leftarrow p_j$
$\tilde{H} \leftarrow H^{(0)}$
Compute the Lagrange multipliers $\tilde{\sigma}_{FX}$ for the active bounds from (2.11)
if $\tilde{\sigma}_{FX} \geq 0$, **then**
  $\tilde{\gamma} \leftarrow 0, \quad u \leftarrow 0$
else

  $\boxed{\text{Compute-descent-direction}}$

end

Figure 2.4: Algorithm IDQP.

2.2.2 Adjusting the penalty parameter

We return to a feature of the merit function in the outer SQP algorithm, the penalty parameter. Let $p = \tilde{p} + \tilde{\gamma} u$. The penalty parameter is adjusted as necessary at each iteration to ensure (p, ξ, q) is a descent direction for the merit function, i.e.,

$$\phi'(0) \leq -\omega, \tag{2.20}$$

Subroutine Move-to-stationary-point

repeat

$g_j \leftarrow \bar{P}_j^T(g + H^{(0)}p_j)$

stationary_point $\leftarrow Z_j^T g_j = 0$

if not *stationary_point* **then**

Solve for d satisfying $\begin{pmatrix} H_j^{(0)} & A_j^T \\ A_j & 0 \end{pmatrix} \begin{pmatrix} d \\ -\pi \end{pmatrix} = \begin{pmatrix} -g_j \\ 0 \end{pmatrix}$

$\bar{d} \leftarrow P_j^T \begin{pmatrix} d \\ 0 \end{pmatrix}$

$\gamma_M \leftarrow \min_\tau \left\{ \dfrac{x_\tau + p_\tau}{|\bar{d}_\tau|} \mid \bar{d}_\tau < 0 \right\}$

hit_constraint $\leftarrow \gamma_M < 1$

$\gamma \leftarrow$ **if** *hit_constraint* **then** γ_M **else** 1

$p_{j+1} \leftarrow p_j + \gamma \bar{d}$

Update working set and compute P_{j+1}, A_{j+1}, Z_{j+1}, and $H_{j+1}^{(0)}$

$j \leftarrow j + 1$

end

until *stationary_point*

Figure 2.5: Subroutine Move-to-stationary-point.

Subroutine Compute-descent-direction

Compute $\hat{u}$ that satisfies: $\begin{pmatrix} \tilde{H} & A^T & I \\ A & 0 & 0 \\ \widehat{P}^T & 0 & 0 \end{pmatrix} \begin{pmatrix} \hat{u} \\ -\tilde{\pi} \\ -\hat{\sigma} \end{pmatrix} = \begin{pmatrix} -\tilde{g} \\ 0 \\ \tilde{\sigma}_{FX}^{-} \end{pmatrix}$

Let γ_1 denote the largest feasible step from $\tilde{p}$ along $u \equiv \hat{u}/\|\hat{u}\|$

if $-(g + \tilde{H}\tilde{p})^T u < \gamma_1 u^T \tilde{H} u$, **then**

$\tilde{\gamma} \leftarrow \min\Big(-\dfrac{(g + \tilde{H}\tilde{p})^T u}{u^T \tilde{H} u}, \gamma_M\Big)$

else

$\tilde{\gamma} \leftarrow \min(\gamma_1, \gamma_M)$

end

if $\|\tilde{p} + \tilde{\gamma} u\| < \|\tilde{p}\|$, **then** $\tilde{\gamma} \leftarrow 0$

if $g^T u > 0$, **then** $\tilde{\gamma} \leftarrow 0$

Figure 2.6: Subroutine Compute-descent-direction.

where

$$\phi'(0) = g^T p + (2\lambda - \mu)^T (c - s) - \rho \|c - s\|^2 \tag{2.21}$$

and

$$\omega = \frac{1}{2}\left((\tilde{p} - p_0)^T \tilde{H} (\tilde{p} - p_0) + \|c - s\|^2 - 2\tilde{\gamma} g^T u \right). \tag{2.22}$$

If (2.20) does not hold with the penalty parameter from the previous iteration, ρ is modified to satisfy

$$\rho \geq \frac{\frac{1}{2}\omega + g^T p + (2\lambda - \mu)^T (c - s)}{\|c - s\|^2}. \tag{2.23}$$

If $\|c - s\| = 0$, then $p_0 = 0$ and $g^T \tilde{p} + \frac{1}{2}\tilde{p}^T \tilde{H} \tilde{p} \leq 0$, implying

$$\phi'(0) = g^T \tilde{p} + \tilde{\gamma} g^T u \leq -\frac{1}{2}\tilde{p}^T \tilde{H} \tilde{p} + \tilde{\gamma} g^T u = -\omega,$$

so no adjustment to ρ is necessary.

2.2.3 Statement of the algorithm

Given x_0, g_0, A_0, c_0, B_0, and λ_0, select $\rho_{-1} \geq 0$, $0 < \sigma < \frac{1}{2}$, $\frac{1}{2} < \eta < 1$, $\beta_c \geq \|c^-(x_0)\|_\infty$, $\beta_\mu \geq \|\lambda_0\|$, $\beta_H > 1$ and $\beta_\rho > 0$. The algorithm IDSQP is presented in Figure 2.7.

2.2.4 Convergence results for the indefinite SQP algorithm

For their second-derivative algorithm SQP2D, Murray and Prieto prove global convergence to a KKT point satisfying the second-order optimality conditions, quadratic convergence of both x_k and λ_k, and boundedness of the penalty parameter [MP99] (see Theorems 3.3, 4.1, and 4.2).

Under weaker assumptions, weakened versions of these results carry over to IDSQP. In particular, for any indefinite quasi-Newton Hessian approximation satisfying **HC3**, the same proofs imply IDSQP has global convergence to a KKT point at a superlinear rate [MP99].

Assumptions

The proofs of convergence that apply to IDSQP permit the stationary point $\tilde{p}$ to be any constrained stationary point encountered by the active-set method, not only the first stationary point, as long as "the reduced Hessians of the modified QP are sufficiently positive

Algorithm IDSQP

$k \leftarrow 0$
repeat
  $H \leftarrow B_k$
  Obtain a feasible step p_0, a QP stationary point $\tilde{p}$,
    a modified Hessian $\tilde{H}$, a sufficient descent direction u,
    and a feasible steplength $\tilde{\gamma}$ in the direction u from $\tilde{p}$,
    by calling IDQP on the QP subproblem

$$\begin{array}{ll} \min_p & g_k^T p + \frac{1}{2} p^T H p \\ \text{s.t.} & A_k p + c_k \geq 0 \end{array}$$

  $\bar{B}_k \leftarrow \tilde{H}$
  $p_k \leftarrow \tilde{p} + \tilde{\gamma} u$
  Optimize the slack variables s_k (cf. (2.3))
  Form search direction for slacks $q_k \leftarrow A_k p_k + c_k - s_k$
  Form μ_k, an estimate of λ^*
  Form multiplier search direction $\xi_k \leftarrow \mu_k - \lambda_k$
  $\omega \leftarrow \frac{1}{2}((\tilde{p} - p_0)^T \bar{B}_k (\tilde{p} - p_0) + \|c_k - s_k\|^2 - 2\tilde{\gamma} g_k^T u)$
  if $\phi_k'(0) \leq -\frac{1}{2}\omega$
    $\rho_k \leftarrow \rho_{k-1}$
  else

$$\rho_k \leftarrow \max\left(2\rho_{k-1}, \frac{\frac{1}{2}\omega + g_k^T p_k + (2\lambda_k - \mu_k)^T (c_k - s_k)}{\|c_k - s_k\|^2}, \beta_\rho\right)$$

  end
  Perform a linesearch on the merit function to choose α_k (Figure 2.1).

$$\begin{pmatrix} x_{k+1} \\ \lambda_{k+1} \end{pmatrix} \leftarrow \begin{pmatrix} x_k \\ \lambda_k \end{pmatrix} + \alpha_k \begin{pmatrix} p_k \\ \xi_k \end{pmatrix}$$

  Evaluate g_{k+1}, A_{k+1} and c_{k+1}
  Update B_k to form B_{k+1}
  $k \leftarrow k + 1$
until *converged*

Figure 2.7: Algorithm IDSQP.

definite on all the subsets of constraints encountered when determining a stationary point." This requirement will be explored further in Chapter 5.

All assumptions made for PDSQP (**A1**-**A7**, **MC1**-**MC3**, **HC1**-**HC3**) in Section 2.1.4 are made here again, with one exception. We no longer assume the Hessian approximation is positive definite, but we do assume the matrix has bounded norm, that is, its eigenvalues are bounded above and below. Assumption HC2 is replaced by

HC2b. There exists $\beta_{lvB} > -\infty$ such that $\forall k$, the eigenvalues of B_k are bounded below by β_{lvB}.

Chapter 3

Direct Hessian Approximation

3.1 Introduction

In this chapter, we describe how the quasi-Newton approach to approximating the Hessian of the Lagrangian in SQP is most commonly implemented, using the BFGS update and a single, positive-definite approximation matrix. We refer to this approach as the *direct* approximation to the Hessian of the Lagrangian. This approach is analyzed and its various drawbacks are discussed, mainly in the context of how they affect the convergence rate, storage, and computation time of the SQP algorithm.

A close relationship exists between the choice of Hessian in the QP subproblem and the convergence rate of the SQP algorithm. Under certain assumptions, when the QP Hessian is the exact Hessian of the Lagrangian, the SQP method can be shown to attain a quadratic rate of convergence in the neighborhood of the solution. In practice, however, the Hessian of the Lagrangian is often approximated from first derivatives, leading to a deterioration in the convergence rate of the SQP algorithm.

3.2 Overview of direct approximation

As mentioned in Chapter 1, the quasi-Newton approach was developed originally for unconstrained and linearly constrained optimization. In the nonlinearly constrained case, we need to estimate the Hessian of a composite function including the objective function, constraint functions, and Lagrange multipliers.

The direct approach is initialized with an approximation matrix B_0, typically the identity matrix. In the unconstrained case, this makes p_0 the steepest descent direction. Alternatively, one pre-scales the first iteration by setting the initial matrix B_0 to a positive diagonal matrix or any positive-definite matrix.

At each iteration, a quasi-Newton update is performed that satisfies condition (1.14). Usually this is the BFGS update

$$\bar{B} = B - \frac{B_k \delta_k \delta_k^T B_k}{\delta_k^T B_k \delta_k} + \frac{y_k y_k^T}{y_k^T \delta_k}, \tag{3.1}$$

but even if not, the update clearly depends on the gradient difference of the Lagrangian y_k, a function of x_k and two, possibly different, multiplier estimates $\widehat{\pi}$ and $\bar{\pi}$:

$$y_k = \nabla_x \mathcal{L}(x_{k+1}, \widehat{\pi}) - \nabla_x \mathcal{L}(x_k, \bar{\pi}). \tag{3.2}$$

At each iteration a current multiplier estimate π_{k+1} is updated from π_k. To compute y_k accurately, it is necessary to decide which multiplier estimates to use at each iteration and to consider how the multiplier estimates themselves are functions of x. Incidentally, initial multiplier estimates are unnecessary because on the first iteration y_1 can be computed using only the first multiplier estimates π_1 for both $\widehat{\pi}$ and $\bar{\pi}$.

SNOPT [GMS97] bases the QP Hessian not on the Lagrangian but on a related function called the *modified Lagrangian*. In Section 3.3 we describe the modified Lagrangian and give reasons for its use.

In Section 3.4 we discuss the multiplier estimates and the definition of the gradient of the Lagrangian to be used in defining the quasi-Newton condition. In Section 3.5 we describe how the BFGS update is modified to preserve positive definiteness.

Large-scale approaches are described in Section 3.6. An overall analysis of this approach is discussed in Section 3.7.

3.3 Modified Lagrangian

The *modified Lagrangian* is

$$\mathcal{L}(x, x_k, \pi_k) = f(x) - \pi_k^T c(x) + \pi_k^T c_L(x, x_k), \tag{3.3}$$

where c_L is the *constraint linearization*:

$$c_L(x, x_k) \quad = \quad c(x_k) + A(x_k)(x - x_k);$$

see Robinson [Rob72], Van der Hoek [Van82], and Gill *et al.* [GMS97].

The first and second derivatives of the modified Lagrangian with respect to x are

$$\begin{aligned} \nabla \mathcal{L}(x, x_k, \pi_k) &= g(x) - (A(x) - A(x_k))^T \pi_k, \\ \nabla^2 \mathcal{L}(x, x_k, \pi_k) &= \nabla^2 f(x) - \sum_i (\pi_k)_i \nabla^2 c_i(x). \end{aligned}$$

Observe that $\nabla^2 \mathcal{L}$ is independent of x_k (and is the same as the Hessian of the conventional Lagrangian). At $x = x_k$, the modified Lagrangian has the same function and gradient values as the objective:

$$\mathcal{L}(x_k, x_k, \pi_k) = f(x_k), \qquad \nabla \mathcal{L}(x_k, x_k, \pi_k) = g(x_k).$$

The first-order term of the QP objective is the linear approximation $g(x_k)^T p$, where $g(x) = \nabla f(x)$ rather than $g(x) = \nabla \mathcal{L}(x, \pi)$.

3.3.1 Benefits of using the modified Lagrangian

There are several benefits to modeling the quadratic objective on the modified Lagrangian. First, when the quadratic objective of the subproblem is modeled after the pure Lagrangian, the multipliers of the subproblem do not approximate the multipliers of the main problem. Instead, the QP multipliers approximate the error in the previous multiplier estimate. In the best case the error converges to zero, but this may cause numerical difficulties when solving the QP subproblem. On the other hand, if the modified Lagrangian is used, the multipliers of the QP subproblem do approximate the multipliers of the original problem.

Using the modified Lagrangian does not affect the minimizer of the QP subproblem. Moreover, at the solution ($x_k = x^*$ and $\pi_k = \pi^*$), the optimality conditions of the QP subproblem match the optimality conditions of the original problem, so the multipliers of the QP subproblem at the solution are π^*. For these reasons and others, the modified Lagrangian was incorporated in SNOPT [GMS97].

3.4 Multiplier estimates and the quasi-Newton update

Since our goal is for the Hessian approximation to converge to a matrix related to the Hessian of the Lagrangian at x^* (i.e., for the BTW condition (1.9) to be satisfied), if it were possible we would set both $\widehat{\pi}$ and $\bar{\pi}$ in (3.2) to optimal Lagrange multipliers π^*. In practice, π^* is not known but is approximated at each iteration by π_k in such a manner that if x_k converges to a solution, then π_k converges to π^*. Potential sources for the multiplier estimates π_k include

- the QP multipliers,
- the least-squares multipliers (see, e.g., [GM79]), or
- the multiplier estimates used in the merit function and updated at each iteration via linesearch.

The typical procedure for computing y_k in the direct approximation of the Hessian assumes $\widehat{\pi}$ and $\bar{\pi}$ are equal and constant with respect to x (that is $\pi = \widehat{\pi} = \bar{\pi}$ and $\partial\pi/\partial x = 0$), and y_k from (3.2) simply reduces to

$$\begin{aligned} y_k &= \nabla\mathcal{L}(x_{k+1},\pi) - \nabla\mathcal{L}(x_k,\pi) \\ &= g_{k+1} - g_k - (A_{k+1} - A_k)^T\pi. \end{aligned} \tag{3.4}$$

Of course, π^* is approximated by varying estimates π_k that are updated at every iteration. The question arises whether a single estimate should be used for $\widehat{\pi}$ and $\bar{\pi}$ (e.g., both π_k or both π_{k+1}) or whether different values should be used in each gradient term of (3.4). Each of these choices has questionable validity because π is no longer constant between iterations (as was assumed); in one case, it is not even constant within the iteration. Despite these reservations, three obvious choices for y_k are

$$\begin{aligned} y_k &= \nabla\mathcal{L}(x_{k+1},\pi_{k+1}) - \nabla\mathcal{L}(x_k,\pi_k) \\ &= (g_{k+1} - g_k) - (A_{k+1} - A_k)^T\pi_{k+1} + A_k^T(\pi_k - \pi_{k+1}), \end{aligned} \tag{3.5}$$

$$\begin{aligned} y_k &= \nabla\mathcal{L}(x_{k+1},\pi_{k+1}) - \nabla\mathcal{L}(x_k,\pi_{k+1}) \\ &= g_{k+1} - g_k - (A_{k+1} - A_k)^T\pi_{k+1}, \end{aligned} \tag{3.6}$$

$$\begin{aligned} y_k &= \nabla\mathcal{L}(x_{k+1}, \pi_k) - \nabla\mathcal{L}(x_k, \pi_k) \\ &= g_{k+1} - g_k - (A_{k+1} - A_k)^T \pi_k. \end{aligned} \tag{3.7}$$

In (3.5), y_k incorporates two different values of π. This formula may be the easiest to implement, since $\nabla\mathcal{L}(x_k, \pi_k)$ can be computed at iteration k and $\nabla\mathcal{L}(x_{k+1}, \pi_{k+1})$ can be computed at iteration $k+1$, each with the latest values of x and π. If the multiplier estimates are nearly equal ($\pi_{k+1} \approx \pi_k$), the final term $A_k^T(\pi_k - \pi_{k+1})$ of (3.5) drops off, and the decision to use different values of π has little effect. More generally, however, we expect $\pi_{k+1} \not\approx \pi_k$, so the final term $A_k^T(\pi_k - \pi_{k+1})$ could contribute significantly to (3.5). This is essentially an error term that would not be present if the constant π assumption were satisfied.

Consider next (3.6), where the newer multiplier estimate π_{k+1} is used for both gradient evaluations. When only one multiplier estimate is used, the term corresponding to $A_k^T(\pi_k - \pi_{k+1})$ in (3.5) disappears. The third case (3.7) also has no such error term. We conclude that using the same multiplier estimate in both terms is preferable, although in all cases the constant π assumption is violated.

3.4.1 Multiplier estimates with the modified Lagrangian

Surprisingly, the decision to use the same or different multiplier estimates can be avoided altogether by using the modified Lagrangian instead of the Lagrangian. Applying the quasi-Newton condition to the modified Lagrangian simplifies the choice of $\widehat{\pi}$ and $\bar{\pi}$ because y_k depends solely on the multiplier estimate used in the first term, $\widehat{\pi}$. The obvious thing is to let $\widehat{\pi} \equiv \pi_{k+1}$, and it is irrelevant what is used for $\bar{\pi}$, since

$$\begin{aligned} y_k &= \nabla\mathcal{L}(x_{k+1}, x_k, \pi_{k+1}) - \nabla\mathcal{L}(x_k, x_k, \bar{\pi}) \\ &= g_{k+1} - g_k - (A_{k+1} - A_k)^T \pi_{k+1}. \end{aligned} \tag{3.8}$$

The Hessian of the modified Lagrangian is independent of the second parameter x_k. However, it is important that the same x_k be used in both terms of (3.8). If y_k were computed as

$$\begin{aligned} y_k &= \nabla\mathcal{L}(x_{k+1}, x_{k+1}, \pi_{k+1}) - \nabla\mathcal{L}(x_k, x_k, \pi_k) \\ &= g_{k+1} - g_k, \end{aligned} \tag{3.9}$$

then the update would only take into account the curvature of the objective function, implying that the resulting quasi-Newton approximation would have the curvature of $f(x)$ only.

3.5 Modifications to maintain positive definiteness

Powell emphasized in [Pow77] the importance of maintaining positive definiteness in quasi-Newton methods for constrained optimization. First, keeping the matrix $B_k \succ 0$ ensures that a descent direction to a suitable merit function can be found from the QP subproblem. Second, Powell observed that positive definiteness propagates a feature of the BFGS method for unconstrained optimization, namely, that the method is invariant under a linear transformation of the variables. The method could not be guaranteed invariant if the solution to the subproblem were not unique.

Furthermore, as long as it maintains positive definiteness, the BFGS update has the special property that it is the unique update satisfying the quasi-Newton condition that minimizes $\|E_k\|_F$, the Frobenius norm of the modification E_k (1.10).

The BFGS update maintains positive definiteness if and only if the approximate curvature $y_k^T\delta_k$ is positive. When there are no constraints, it is possible to terminate the linesearch so that this condition holds. In the nonlinear inequality constrained case, it is not possible to define linesearch termination conditions that ensure $y_k^T\delta_k > 0$. Since the Hessian of the Lagrangian need not be positive definite at a local minimizer, the approximate curvature $y_k^T\delta_k$ can be negative or very small at points arbitrarily close to (x^*, π^*).

When $y_k^T\delta_k$ is negative or nearly zero, performing the BFGS update leads to an indefinite or poorly conditioned Hessian approximation. On the other hand, if the update is skipped when $y_k^T\delta_k < 0$, new information about curvature of the Lagrangian is ignored. It has been observed in practice that skipped updates causes the convergence rate of the SQP algorithm to deteriorate.

3.5.1 Powell's modification

To deal with this difficulty, Powell suggested that whenever the curvature is not sufficiently positive, y_k be replaced by a convex combination of y_k and $B_k\delta_k$. If the BFGS update (1.15) was made properly at the last iteration, then $B_k\delta_k = y_{k-1}$. Powell's test for positive curvature is

$$y_k^T \delta_k \geq 0.2\ \delta_k^T B_k \delta_k, \tag{3.10}$$

where 0.2 is an empirically chosen constant (see [Pow77, Pow78a, Pow78b]). When this test fails, Powell replaces y_k in the BFGS update (3.1) by

$$\eta_k = \theta y_k + (1-\theta) B_k \delta_k, \quad 0 \leq \theta \leq 1. \tag{3.11}$$

The parameter θ is chosen so that positive curvature is attained, i.e.,

$$\eta_k^T \delta_k = \theta y_k^T \delta_k + (1-\theta) \delta_k B_k \delta_k \geq 0.2\ \delta_k^T B_k \delta_k. \tag{3.12}$$

This condition is clearly attainable at $\theta = 0$ since $\eta_k^T \delta_k = \delta_k^T B_k \delta_k$. But setting θ to zero is undesirable because it would be equivalent to skipping the update.

Of the possible values that satisfy (3.12), Powell selects $\theta \in [0,1]$ to minimize $\|\eta_k - y_k\|$. By this measure the optimal value of θ is

$$\theta_k = \frac{0.8\ \delta_k^T B_k \delta_k}{\delta_k^T B_k \delta_k - y_k^T \delta_k}.$$

Incidentally, this is also the maximum value of θ to satisfy the criterion for positive curvature (3.12). Substituting θ_k back into the formula for η_k, we see that the curvature of the modified update is exactly equal to $0.2 \delta_k^T B_k \delta_k$, independent of the amount of negative curvature $y_k^T \delta_k$.

3.5.2 Approach used in SNOPT

The SQP algorithm implemented within SNOPT uses an elaborate procedure to alter the update [GMS97] when the curvature is considered not sufficiently positive. SNOPT performs the matrix update based upon the modified Lagrangian, with δ_k and y_k defined as

$$\delta_k = x_{k+1} - x_k, \qquad y_k = \nabla \mathcal{L}(x_{k+1}, x_k, \pi_{k+1}) - \nabla \mathcal{L}(x_k, x_k, \pi_{k+1}). \tag{3.13}$$

The update is modified when $y_k^T \delta_k$ does not exceed some positive threshold. Two attempts are made to modify the update: first, modifying δ_k and y_k, and second, modifying only y_k. If neither modification provides sufficiently positive approximate curvature, no update is made.

First modification

This procedure attempts to make $y_k^T\delta_k$ sufficiently positive by substituting a new point z_k for the previous iterate x_k when computing y_k and δ_k. One disadvantage is that additional work is required; in order to apply formula (3.13), it is necessary to reevaluate the functions and their gradients at the newly defined point z_k.

The rationale behind the first modification is to seek a subspace in which the curvature is positive and to use this information to alter the update. Such a subspace is suggested by the properties of the reduced Hessian at a local minimizer of NP.

Before defining z_k, we identify several other terms. Let $\widehat{x}_k$ be the solution of the QP subproblem. The update x_{k+1} to x_k depends on a stepsize α_k in the manner

$$x_{k+1} = x_k + \alpha_k p_k,$$

where

$$p_k = \widehat{x}_k - x_k.$$

The range-space and null-space portions of the QP direction p_k will be denoted p_R and p_N. Let $\bar{x}_k$ be the first feasible iterate found for the QP subproblem.

We now define the new point z_k as

$$z_k = x_k + \alpha_k(\bar{x}_k - x_k) = x_k + \alpha_k p_R.$$

The corresponding values for δ_k and y_k are

$$\delta_k = x_{k+1} - z_k = \alpha_k p_N, \quad \text{and} \qquad y_k = \nabla\mathcal{L}(x_{k+1}, x_k, \pi_{k+1}) - \nabla\mathcal{L}(z_k, x_k, \pi_{k+1}).$$

Substituting for δ_k and recognizing that $y_k \approx \nabla^2\mathcal{L}(x_k, x_k, \pi_k)\alpha_k p_N$, we have

$$y_k^T\delta_k = \alpha_k y_k^T p_N \approx \alpha_k^2 p_N^T \nabla^2\mathcal{L}(x_k, x_k, \pi_k) p_N.$$

Hence, $y_k^T\delta_k$ approximates the curvature along p_N. If p_R is small compared with p_N, it follows that $y_k^T\delta_k$ approximates the curvature for the reduced Hessian, which must be positive semi-definite at a minimizer of NP.

Second modification

If the first modification attempt fails, a second approach is tried in SNOPT. The rationale behind the second modification is that we may replace the Hessian of the (modified) Lagrangian by the Hessian of an augmented Lagrangian without impacting the rate of convergence. Indeed, this is the basis of the proof by Powell that an SQP algorithm with a positive-definite Hessian approximation has the potential for superlinear convergence even when the Hessian of the Lagrangian is indefinite.

For this modification, choose Δy_k so that $(y_k + \Delta y_k)^T \delta_k = \sigma_k$ (if possible), and redefine y_k as $y_k + \Delta y_k$. To obtain Δy_k, consider the *augmented* modified Lagrangian [MS82]:

$$\mathcal{L}_A(x, x_k, \pi_k) = f(x) - \pi_k^T d_L(x, x_k) + \frac{1}{2} d_L(x, x_k)^T \Omega d_L(x, x_k), \tag{3.14}$$

where Ω is a matrix of parameters to be determined: $\Omega = \mathrm{diag}(\omega_i)$, $\omega_i \geq 0$, $i = 1, \ldots, m$. The perturbation

$$\Delta y_k = (A(x_{k+1}) - A(x_k))^T \Omega d_L(x_{k+1}, x_k) \tag{3.15}$$

is equivalent to redefining the gradient difference as

$$y_k = \nabla \mathcal{L}_A(x_{k+1}, x_k, \pi_{k+1}) - \nabla \mathcal{L}_A(x_k, x_k, \pi_{k+1}). \tag{3.16}$$

Choose the smallest (minimum two-norm) ω_i's that increase $y_k^T \delta_k$ to σ_k. They are determined by the linearly constrained least-squares problem

LSP	$\underset{\omega}{\text{minimize}} \quad \|\omega\|^2$
	subject to $a^T \omega = \beta, \quad \omega \geq 0,$

where $\beta = \sigma_k - y_k^T \delta_k$ and $a_i = v_i w_i$ $(i = 1, \ldots, m)$, with $v = (A(x_{k+1}) - A(x_k))\delta_k$ and $w = d_L(x_{k+1}, x_k)$. The optimal ω can be computed analytically [GMSW86, Eld91]. If no solution exists, or if $\|\omega\|$ is very large, the second modification attempt fails and no update is made.

3.6 Computational cost and scalability

Dense quasi-Newton updates require $O(n^2)$ multiplications per iteration (for a matrix of size n) and $O(n^2)$ storage space. The BFGS update in particular requires $4n^2 + 2n$ multiplications per iteration. Its memory requirement is $\frac{1}{2}n^2 + 2n$ locations to store B_k, y_k, and δ_k.

Large-scale optimization Since both storage requirements and computational time grow quadratically in n, special care is taken to reduce the storage and effort required to perform the BFGS update on large-scale problems.

Much of the work done on extending quasi-Newton methods to large problems has been directed at the unconstrained case. Research in this area may be classified under the following general approaches:

(i) Limited-memory approximation.

(ii) Sparse quasi-Newton approximation.

(iii) Reduced Hessian approximation.

(iv) Partial separability.

Limited-memory approximations In this sparse method, the BFGS approximation matrix B_k is not stored *explicitly*; rather, the initial approximation is stored (usually a diagonal matrix) together with the information required to perform the updates. Since each update is a rank-two matrix, at most two vectors and two constants are required per update. Instead of B_k being a result of k updates, only a limited number of updates are used. There are various ways to implement this approach. We could, for instance, keep the last t updates, where t is small, say less than 25 and typically less than ten. Depending on the form of the update, the storage requirements are at worst $2nt$ and at best nt.

Limited-memory approximations have been the subject of considerable research and experimentation (see [Noc80, ZBLN97, BL85, GL89, BNS94, Leo95]). Much of the work on limited-memory methods has been directed at the unconstrained case. Although the concept is simple, there are many ways to implement a limited-memory method. How the updates are stored and in what form impacts the cost of operations, such as matrix-vector multiplications, with B_k. Since B_k is not stored explicitly this limits the methods of solution of the QP subproblems.

The limited-memory approach has the property that it does not depend on the Hessian of the Lagrangian being sparse. The hope is that a good approximation of the Hessian is not critical to success provided a good approximation of the *reduced* Hessian is obtained. The potential exists for a good approximation after a small number of iterations when the dimension of the reduced Hessian is much smaller than that of the Hessian.

Sparse quasi-Newton The success of quasi-Newton methods on small dense unconstrained problems prompted research on methods that could update sparse approximations having the same sparsity pattern as the exact Hessian. For constrained problems we are concerned with the sparsity pattern of $\nabla_x^2 \mathcal{L}(x, \lambda)$. Despite considerable work, few useful results have emerged, especially for constrained optimization. Moreover, it is computationally costly to perform the updating. A key difficulty is to preserve the correct sparsity pattern and the property of hereditary positive definiteness. This approach determines an explicit representation of B. Hence, it is only applicable to problems for which W is sparse.

Reduced-Hessian approximation Given a nonsingular matrix Q_k and an approximation $Q_k^T B_k Q_k$ to the projected Hessian $Q_k^T \nabla^2 \mathcal{L}(x_k) Q_k$, it is not necessary to know B_k explicitly to implement an SQP method, provided Q_k is chosen appropriately and a null-space method is used to solve the QP subproblem.

The *reduced Hessian approach* stores an approximation to the reduced Hessian instead of the larger full Hessian (see [Eld91, BNS95]). The reduced-Hessian approach is likely to be successful whenever the number of constraints active at the solution is similar to the number of variables. For efficiency, the difference may be anything from zero to a few hundred. This is known to be true for many important problems, such as trajectory optimization and process control. Indeed, it is likely to be true for most control problems, since the number of control variables is often small compared to the number of state variables.

Partial separability An approach advocated for determining a compact Hessian approximation for unconstrained optimization problems is to use the property of *partial separability* [GT82a, GT82b]. A function $f(x)$ is *partially separable* if it can be written as a sum of smaller "element functions" $f_i(x)$, that is,

$$f(x) = \sum_{i=1}^{m} f_i(x).$$

Given the structure of f, one can approximate $\nabla^2 f(x)$ by a *partitioned* quasi-Newton update. Storing a separate BFGS approximation for each of the *element Hessians* $\nabla^2 f_i(x)$ provides a sparse representation of $\nabla^2 f(x)$ if each of the f_i involves a (small) subset of the variables x. If $\nabla^2 f(x)$ is not sparse, it need never be stored explicitly. Application of partial separability generally requires the user to provide information describing the structure of the problem. The idea of applying partial separability to approximating the Hessian of the Lagrangian was first suggested by Griewank and Toint in [GT82b]. This application of partial separability would be easy to implement because the structure of the Lagrangian is already known.

Disaggregation This is a special case of partial separability in which the precise structure of the compound function (the Lagrangian) is known and for which the nature of the dependence on the constraint functions may be deduced directly from the sparsity pattern of the Jacobian. The basic assumption being made is that the largest rank of the individual Hessians is small. This approach is explored in the next chapter.

3.7 Analysis of the direct BFGS approximation

Our analysis begins by examining the effect of the direct BFGS approximation on the convergence rate of the SQP algorithm. We also discuss the memory and computation time per iteration required by the estimates.

The BTW sufficient condition (1.9) for superlinear convergence was presented in the introduction. For general functions, it is unknown whether the Hessian approximations computed by the direct BFGS update ever converge, let alone satisfy this condition. Even for the case where f and c are quadratic functions, if the Hessian is not assumed to be positive definite, no one has been able to show (1.9) holds for the BFGS update.

Two features of the direct BFGS approximation described in the previous section detract further from the performance of the SQP algorithm:

1. Adjustments made to the BFGS update to preserve positive definiteness.

2. Fluctuating multiplier estimates and different choices for y_k.

3.7.1 Positive definiteness

Maintaining positive-definite approximations to the Hessian of the Lagrangian has certain disadvantages. Most importantly, as mentioned earlier, the true Hessian of the Lagrangian is not guaranteed to be positive definite. Even in the neighborhood of a minimizer, the Lagrangian need only have positive curvature in the null-space of the active constraints. As demonstrated by Powell's modification to BFGS (3.11), the assumption of positive definiteness makes the BFGS update (3.1) difficult to perform.

Powell's modification has other disadvantages as well. First, the modified BFGS update no longer minimizes the Frobenius norm. Second, Powell's modified update no longer satisfies the quasi-Newton condition. Despite this, Powell showed it was possible for an SQP algorithm with a positive-definite Hessian approximation to give superlinear convergence even when the exact Hessian of the Lagrangian is indefinite. He concluded that his modified method could still attain superlinear convergence, although superlinear convergence is not guaranteed, even for quadratic functions. Implementing the two SNOPT modifications is even more complicated than Powell's modification but has the same drawbacks. It is not guaranteed to work, and the update no longer satisfies the quasi-Newton condition.

In addition, positive definiteness is incompatible in practice with maintaining sparsity. Limited-memory methods applied to large-scale problems ignore the sparsity structure of the problem.

3.7.2 Invariance under linear transformation

In other ways, Powell's modification is robust. In particular, Powell's BFGS modification and his test for positive curvature are both invariant under linear transformation.

Consider a linear transformation of variables $y = Ax$. It is desirable that an SQP algorithm be invariant under such a transformation for several reasons. For one, the initial scaling of the variables would not affect the outcome of the algorithm. Second, algorithms that are invariant behave better on poorly conditioned problems. Newton's method is invariant under linear transformation while steepest descent is not. It may be true that an algorithm is invariant under scaling, even if not under a more general linear transformation. See [Fle81] for further background.

All Broyden family quasi-Newton updates for *unconstrained* optimization (including BFGS) are invariant under linear transformation as long as the initial Hessian approximation is also appropriately transformed. For a linesearch method to be invariant, the linesearch termination criterion, which determines the stepsize, must also be invariant. The linesearch termination criterion is invariant for the unconstrained BFGS method as long as the termination criterion depends only on $g^T p$.

For constrained optimization methods such as SQP, there are additional requirements for invariance under linear transformation. The Hessian update needs to be invariant, the search direction found from the subproblem needs to be invariant (which would present a problem if the QP subproblem did not have a unique solution), and the termination criterion of the linesearch (for SQP methods usually based on a merit function) is required to be invariant.

3.7.3 Interpretation of the different choices for y_k

Although the formulas for y_k (3.5)–(3.7) were designed under the assumption that π is a constant (see (3.4)), certain weaker assumptions would justify these formulas for the quasi-Newton update even if the multiplier estimates vary from iteration to iteration. Such weaker assumptions still rely on a consistent value for π *within* the iteration; it is not possible to justify using different estimates of π in each term of y_k, as in (3.5).

Some additional notation needs to be introduced. Let $B_k(\pi_k)$ be the matrix at iteration k that approximates the Hessian of the Lagrangian $\nabla^2\mathcal{L}(x_k, \pi_k)$. Let $B_{k+1}(\pi_{k+1})$ be the updated matrix to approximate $\nabla^2\mathcal{L}(x_{k+1}, \pi_{k+1})$.

Weaker assumptions that justify updating $B_k(\pi_k)$ via the formulas for y_k in (3.6) and (3.7) are as follows:

- For y_k based on the newer π_{k+1} in (3.6), one needs to assume

$$B_k(\pi_k) \approx B_k(\pi_{k+1}). \tag{3.17}$$

  Applying the quasi-Newton update to $B_k(\pi_{k+1})$ gives $B_{k+1}(\pi_{k+1})$.

- For y_k based on the older π_k in (3.7), one needs to assume

$$B_{k+1}(\pi_k) \approx B_{k+1}(\pi_{k+1}). \tag{3.18}$$

The quasi-Newton update can be applied initially to $B_k(\pi_k)$, letting the parameters $\widehat{\pi}$ and $\bar{\pi}$ both equal π_k (see 3.7). The update is a correct estimate for $B_{k+1}(\pi_k)$, but by assumption (3.18), we have the desired $B_{k+1}(\pi_{k+1})$.

Neither of these assumptions would be needed if the quasi-Newton update treated π as a differentiable function of x, e.g., $\pi_k = \pi(x_k)$ and $\pi_{k+1} = \pi(x_{k+1})$. Let $A(x)$ and $P(x)$ be the Jacobians of $c(x)$ and $\pi(x)$. The generalized formula for the gradient difference is

$$\begin{aligned} y_k &= \nabla_x(\mathcal{L}(x_{k+1}, \pi(x_{k+1})) - \nabla\mathcal{L}(x_k, \pi(x_k)) && (3.19)\\ &= g_{k+1} - g_k - (A_{k+1}^T\pi_{k+1} - A_k^T\pi_k) - (P_{k+1}^T c_{k+1} - P_k^T c_k), && (3.20)\end{aligned}$$

where $A_k = A(x_k)$ and $P_k = P(x_k)$.

This approach has drawbacks, not least of which is that $\pi = \pi(x)$ must be well-defined and differentiable, and computing the derivatives of $\pi(x)$ may be difficult. For the least-squares multiplier estimates it is possible to describe explicitly how π_k varies as a differentiable function $\pi_k(x)$ of x, but $\pi(x)$ is highly nonlinear. The quasi-Newton update works best when the Lagrangian is a quadratic function, and the performance of the SQP algorithm will suffer the more nonlinear it is.

3.7.4 Why does direct BFGS sometimes work well?

Given our criticism of the direct quasi-Newton update, the question arises as to why at least some of the time the SQP method is observed to achieve superlinear convergence when using the direct BFGS approximation.

Consider the issue of fluctuating multiplier estimates. All terms with derivatives $P(x)$ of $\pi(x)$ (see (3.19)) are ignored in the direct BFGS update. In general this should cause problems, but in some cases it can be justified. Omitting $P(x)$ terms would be acceptable if the term $P(x)^T c(x)$ in (3.19) were dominated by the remaining terms in the gradient of the Lagrangian, $g(x) - A(x)^T\pi(x)$. In particular, if $\|P(x)\|$ were bounded, it would suffice for $\|c\| \ll \|g(x) - A(x)^T\pi(x)\|$, and this is true when the final search directions are in or nearly in the null-space of the constraints. If this is the pattern of convergence near the solution, it may be justified to treat the multiplier estimates as constant.

Under certain favorable conditions the BFGS update need not be modified near the solution, despite the exact Hessian not being positive definite. If the final search directions are in the null-space of the active constraints ($\delta_k = \alpha_k Z_k p_z$), the curvature $y_k^T\delta_k$ is approximately

$$y_k^T\delta_k = \alpha_k y_k^T Z_k p_z \approx \alpha_k^2 p_z^T Z_k^T(\nabla_x^2\mathcal{L}(x_k,\lambda))Z_k p_z.$$

Hence the curvature is nonnegative if the reduced Hessian is positive semi-definite, as at a minimizer of NP. In practice it has been observed that for NPSOL and SNOPT, usually $y_k^T\delta_k > 0$ near the solution [GMSW89, GMS97].

If x_k is in the neighborhood of the solution, if the $\{p_k\}$ are in or are close to the null-space of the active constraints, and if the dimension of the null-space of the active constraints is small as well, then at least the potential exists for an accurate Hessian approximation in the null-space of the constraints after few iterations. For final search directions to be in the null-space of the constraints, it is critical that the unit step be taken in the linesearch and that the Jacobian have full rank at the solution, so that the infeasibilities converge at a quadratic rate. We may expect the direct BFGS update not to work well whenever the Jacobian is ill-conditioned at the solution because we are then no longer assured of positive curvature along the search direction and because the search directions may no longer lie in a small subspace.

To sum up, the direct, positive-definite BFGS Hessian approximation has several shortcomings that reduce the likelihood of satisfying the conditions for superlinear convergence. First, while the BFGS update for unconstrained optimization could be proven to converge to the exact Hessian, the BFGS update for constrained optimization cannot be. Second, the BFGS update enforces positive definiteness, but at a price—one is periodically forced to skip or modify the update. Third, using changing multiplier estimates violates an assumption of the BFGS update and hinders convergence to the exact Hessian.

Chapter 4

Disaggregated Hessian Approximation

4.1 Introduction

Direct, positive-definite, BFGS approximation to the Hessian of the Lagrangian ignores the dependence of the Lagrangian on the multipliers and enforces a positive-definite approximation. For these and other reasons, the direct approach may perform poorly even on ideal problems.

In this chapter we explore alternative ways of approximating the Hessian of the Lagrangian $W(x, \lambda)$ using quasi-Newton methods. The approaches described in this chapter are designed to avoid the shortcomings of the direct, positive-definite, BFGS approximation. We consider strategies to improve the update, which include

1. relaxing positive definiteness,

2. disaggregating the estimation problem into smaller parts, and

3. substituting the symmetric rank-one update for the BFGS update.

We analyze these approaches, compare to the direct approach, and describe the computational cost of implementation. Finally, some computational examples are given.

4.2 Relaxing positive definiteness

Suppose the approximation matrix B_k were allowed to be indefinite. The *indefinite* BFGS update (1.15) could then be applied if $y_k^T\delta_k$ were positive or *negative*. As before, the BFGS update would still be undefined when $y_k^T\delta_k = 0$, and in addition it would be undefined if $\delta_k^T B_k \delta_k = 0$ (before, this term was guaranteed to be positive). The zero-denominator case is not a serious concern, but the indefinite BFGS update needs to be otherwise compared to the other indefinite quasi-Newton updates. Once the assumption of positive definiteness is dropped, the indefinite BFGS update can no longer be proven to minimize the Frobenius norm of the change E_k to B_k. The symmetric rank-one (SR1) quasi-Newton update is attractive for its simplicity and other properties.

4.3 Symmetric rank-one update

The SR1 update is an intuitively preferable choice of quasi-Newton update because at each iteration it imposes a rank-one change that corresponds to the information gained about the curvature of the Lagrangian in only one direction (the search direction). The SR1 update is thus simpler than the rank-two BFGS update, and each update can be stored with a single vector and a constant.

Suggested independently around 1967 by a number of authors, the SR1 update is the unique rank-one update that preserves symmetry and satisfies the quasi-Newton condition.

We first present the SR1 update for the unconstrained case. Suppose we wish to estimate the Hessian of a twice continuously differentiable function $F(x) : I\!R^n \to I\!R$ with $g(x_k) = \nabla F(x_k)$. Let B_k be the approximation matrix for $\nabla^2 F(x_k)$, with $\delta_k = x_{k+1} - x_k$ and

$$y_k = g(x_{k+1}) - g(x_k). \tag{4.1}$$

The SR1 update is

$$B_{k+1} = B_k + \frac{v_k v_k^T}{v_k^T \delta_k}, \tag{4.2}$$

where $v_k = y_k - B_k\delta_k$. It is well-defined if the denominator $v_k^T\delta_k$ is nonzero. (For handling of exceptions, see below.)

In the nonlinearly constrained case, the SR1 update can be extended in a straightforward manner to estimate the Hessian of the Lagrangian. Simply substitute y and δ from (1.12) and (1.16).

Undefined updates with SR1

Like the indefinite BFGS update, the SR1 update (4.2) becomes undefined as certain quantities, in this case $v^T\delta$, approach zero. Moreover, it is ill-advised to complete the update if $v^T\delta$ is small in relative terms. (For simplicity, we omit the subscripts on δ and v.)

A practical algorithm requires safeguards to handle these difficulties. Conn *et al.* [CGT88] suggested that the update be skipped when either of

$$|v^T\delta| \le \epsilon_1 \|\delta\| \|v\|, \tag{4.3}$$

or

$$\|B_{k+1} - B_k\| = \|\frac{vv^T}{v^T\delta}\| > \gamma_1, \tag{4.4}$$

where $\epsilon_1 \in (0,1)$ is a constant (set to 10^{-8}) and $\gamma_1 = 10^8$. Condition (4.3) handles the case where δ and v are nearly perpendicular. Condition (4.4) handles the case where the norm of the update $\|vv^T/v^T\delta\|$ is large (equivalently, $|v^T\delta|$ small relative to $\|vv^T\|$); under this condition if the update were performed the new approximation matrix could immediately become extremely large and close to a rank-one matrix.

Checking condition (4.4) does not, however, altogether prevent the updated matrix from becoming nearly equal to a rank-one matrix after a single update. There is still danger of this happening if $\|vv^T/v^T\delta\|$ is large *relative* to $\|B_k\|$, even if not large in absolute terms. For that reason, we suggest expanding (4.4) to exclude situations where

$$\|B_{k+1} - B_k\| = \|\frac{vv^T}{v^T\delta}\| > \gamma_1 \|B_k\|. \tag{4.5}$$

A simpler alternative is to replace (4.4) by

$$\|B_{k+1} - B_k\| = \|\frac{vv^T}{v^T\delta}\| > \gamma_1 (1 + \|B_k\|). \tag{4.6}$$

Rather than skip the update when the SR1 update is ill-defined, we could modify the SR1 update, just as the BFGS update was modified in certain cases. Because the denominator of the SR1 update differs from the BFGS update, steps that preclude one of the updates may allow the other. On iterations where the SR1 update is ill-defined, a well-defined BFGS update could be substituted for the SR1 update (at the risk of losing theoretical convergence properties for the quasi-Newton updates).

The problem of occasionally needing to skip SR1 updates is less serious and less common than needing to skip or modify BFGS updates in order to maintain positive definiteness for an approximation of the Hessian of the Lagrangian. Admittedly, tests (4.3) and (4.4) could fail even on well-defined and well-conditioned minimization problems. But if this happens at all, it is most likely to happen when the current approximation B_k is already very good, since $v = y - B_k\delta$ goes to zero. Skipping an update under such circumstances is not necessarily bad. This contrasts with the BFGS update, which may need to be modified or skipped anywhere negative curvature is encountered, preventing new information from being incorporated into B_k.

SR1 performance properties

The performance of the SR1 update has been analyzed and compared to the BFGS update in the literature, but the analysis has been limited mainly to unconstrained optimization. The traditional supremacy of the BFGS update was first challenged by Conn, Gould, and Toint [CGT88], who conducted numerical experiments on bound-constrained problems within the trust-region framework showing the SR1 update to be substantially more efficient than any of several other quasi-Newton methods tested, including BFGS, DFP, and PSB. They suggested this behavior could be linked to better convergence of the SR1 matrix updates to the true Hessian.

In theory, BFGS should perform well for unconstrained optimization problems because the usual linesearch termination conditions ensure that $y^T\delta > 0$.

For the SR1 update to perform well, all that is required is that the search directions be independent. A key property of the rank-one update is that for a quadratic function the exact Hessian can be computed exactly in n iterations. This well-known result, which seems to have been proven first by Fiacco and McCormick [FM68], is stated here without proof.

Theorem 4.3.1 *Suppose $F(x)$ is a quadratic function with rank-n Hessian H. If the matrices $\{B_k\}$ obtained using the SR1 update are all well-defined, and if $\delta_0, \delta_2, \ldots, \delta_{n'-1}$ are* *independent directions ($n' < n$), then*

$$B_{n'}\delta_k = y_k, \quad k = 0, \ldots, n'-1,$$

where the SR1 update and y_k are defined in (4.2) and (4.1), respectively.

Let Δ be the $n \times n'$ matrix whose kth column is δ_{k-1}. Since the quasi-Newton condition holds at each update, the theorem implies

$$B_{n'}\,\Delta = H\Delta \quad \text{and} \quad \Delta^T B_{n'}\,\Delta = \Delta^T H\Delta.$$

In other words, the matrix approximation is correct in the subspace spanned by the n' search directions. If the n' search directions span the null-space of the constraints, then

$$Z^T B_{n'} Z = Z^T H Z.$$

Under the above assumptions, the SR1 update "terminates" after n iterations with

$$B_n = H.$$

Exact linesearches are not required for these results. In fact, no property of the search directions is required except independence.

Conn, Gould, and Toint [CGT91] extended this known convergence result for the SR1 matrices on quadratics to the general class of sufficiently smooth nonlinear functions.

Later, Khalfan, Byrd, and Schnabel [KBS93] conducted numerical experiments showing the SR1 update to be competitive with or even more efficient than BFGS in trust-region methods for unconstrained optimization.

4.4 Aggregation/disaggregation model

Direct approximation of the Hessian of the Lagrangian, which combines multipliers and second derivatives into one function, has drawbacks regardless of whether one uses the SR1 update or the BFGS update. In theory, the update treats the Lagrange multipliers as constants, rather than as functions of the current iterate, yet in practice the implementation uses the latest multiplier estimates at each iteration. This has negative implications for both theory and practice. Another significant drawback is that we are unable to incorporate full or partial second-derivative information easily using direct approximation.

Both difficulties are overcome by disaggregating the problem. That is, rather than estimate the Hessian of the Lagrangian as a single entity, we shall estimate $\nabla_x^2 \mathcal{L}(x_k, \lambda)$ by estimating its sub-components and then aggregating to form the Hessian approximation.

Since

$$\nabla^2 \mathcal{L}(x_k, \pi_k) = \nabla^2 f(x_k) - \sum_i (\pi_k)_i \nabla^2 c_i(x_k), \tag{4.7}$$

we estimate

$$B_k = B_k^0 - \sum_i (\pi_k)_i \; B_k^i, \tag{4.8}$$

where B_k^0 is our estimate for $\nabla^2 f(x_k)$, B_k^i for $\nabla^2 c_i(x_k)$, and π_k is the multiplier estimate.

Here we can easily use exact second derivatives for B_k^0 or B_k^i, if they are available. Otherwise the Hessians are approximated directly, using a quasi-Newton update or some other method. The Lagrange multipliers must be estimated whether we have exact second derivatives or not.

Lagrange multiplier estimates

Many choices are involved in forming the multiplier estimates, and several goals must be considered. To prove convergence only requires $\{\pi_k\}$ to be bounded. To show superlinear convergence a necessary condition is $\pi_k \to \pi^*$. We seek an estimate that is easy to compute and that converges to the optimal multipliers π^*. Four possibilities to consider are the following.

1. The QP multipliers: the multipliers at the solution of the most recent QP subproblem.

2. The merit function multipliers: the multiplier estimates used in the linesearch on the merit function.

3. The least-squares multipliers: the value π_k that minimizes $\|A_k^T \pi_k - g_k\|$ (an expensive estimate).

4. The Murray-Prieto [MP99] estimate $\pi_k = \lambda_k - \rho_{k-1}(c_k - s_k)$, where λ_k is the current Lagrange multiplier estimate used in the merit function and ρ_{k-1} is the penalty parameter. When exact derivatives are known, this choice is necessary if we wish to

match negative curvature in the Hessian of the Lagrangian to that of the Hessian of the merit function. When second derivatives are approximated, we shall not normally be interested in negative curvature, so this choice of estimate is not essential.

4.5 Estimating individual constraint Hessians

When exact second derivatives are not available, we approximate the individual constraint Hessians using a quasi-Newton update. Two candidates to consider for this update are BFGS and SR1. Since maintaining positive definiteness is not a concern, and since SR1 is simpler and more intuitive, we favor the SR1 update over BFGS. Theorem 4.3.1 implies in certain situations that the SR1 update will converge to the correct matrix in a finite number of iterations.

The SR1 quasi-Newton update can be easily applied to the estimation of the constraint Hessians $\nabla^2 c_i$. Let the gradient difference for the ith constraint be

$$y_k^i = \nabla c_i(x_{k+1}) - \nabla c_i(x_k), \tag{4.9}$$

and define the residual

$$v_k^i = y_k^i - B_k^i \delta_k, \tag{4.10}$$

where δ_k is the change $x_{k+1} - x_k$. Then the SR1 update for B_k^i is

$$B_{k+1}^i = B_k^i + \frac{v_k^i (v_k^i)^T}{(v_k^i)^T \delta_k}. \tag{4.11}$$

4.6 Initialization

It is not immediately clear how to initialize the Hessian estimates B_0^i. Since the component matrices are not necessarily positive definite, it is unclear whether the identity is a suitable initial approximation. A finite-difference estimate is a good choice but may be prohibitively expensive. On the other hand, we would like to initialize $\{B_0^i\}$ so that the initial reduced Hessian $Z_0^T B_0 Z_0$ is positive definite. Quite how to do this is unclear.

If we set the objective Hessian to the identity (or a positive diagonal) and the constraint Hessians to zero, we can achieve a first iteration result similar to that attained by direct updates. This eliminates the need for initial multiplier estimates. It also ensures a positive-definite initial Hessian (and hence, reduced Hessian).

4.7 Computational cost and scalability

A naive or dense implementation of the disaggregation approach would require storing $m+1$ symmetric matrices of dimension n, where m is the number of nonlinear constraints, for a total of $n^2(m+1)/2$ elements. That storage requirement, $m+1$ times greater than required by the direct Hessian update, could easily be prohibitive. Even if the limited-memory approach is applied to the disaggregation model, retaining the L most recent updates would still require $\frac{1}{2}(m+1)$ times the storage needed for the direct BFGS limited-memory approach.

Fortunately, in many cases sparse techniques can be applied to reduce storage and computational cost dramatically without losing accuracy in the estimate. The sparsity structure of the constraint Hessians can be deduced easily from information already required by codes for large-scale nonlinear constrained optimization such as MINOS [MS78, MS82, MS95] and SNOPT [GMS97]. Large problems are assumed to be sparse by MINOS and SNOPT. With few exceptions, that usually means that the objective and constraint functions are functions of very few variables, typically less than 10. A distinction is made in MINOS and SNOPT between variables that occur nonlinearly (in any constraint or the objective function) and those that occur only linearly, and the nonlinear variables are generally placed first before the linear variables. The Jacobian is input by the user in a compact form representing the values and locations of the nonzero elements. Alternatively, it may be possible to determine derivatives and sparsity patterns automatically using a package for automatic differentiation such as ADIFOR for Fortran or ADOL-C for C/C++. Suppose there are $\widehat{n}$ nonlinear variables. We refer to the subset of the Jacobian in the nonlinear variables only, i.e. the first $\widehat{n}$ columns of the Jacobian, as $J_{\widehat{n}}$.

An efficient sparse representation of the component Hessians can be computed using information already available about the sparsity structure of the Jacobian and gradient. The component Hessian for a given constraint can be represented as a square dense matrix in any superset of the nonlinear variables in that constraint. The nonzero elements of the associated row of $J_{\widehat{n}}$ corresponds to such a superset. A similar result follows for the objective function and gradient. Lagrange multiplier estimates are stored in a separate vector that is updated at every major iteration.

Sparse storage of the disaggregated Hessian of the Lagrangian is much easier to implement than applying partial separability to estimate Hessians of general functions for one important reason. When applying partial separability to general functions, the user needs to provide additional structural information about the problem. There is no such drawback in our case. Although it may still be necessary for the user to provide the sparsity structure of the objective function, this is only a single function and represents a minor inconvenience.

The storage requirements for a sparse representation of the approximation to the Hessian of the Lagrangian are easy to compute. Suppose there are at most t nonzeros in any row of $J_{\widehat{n}}$ or the gradient. It follows from the above discussion that we might reasonably assume t is small ($t \ll n$), i.e., all rows of the Jacobian are sparse. A transformation of variables may lead to even further decrease in the number of nonconstant values in the Jacobian. The individual constraint Hessians are stored and updated as dense symmetric matrices of dimension t. The total number of values that needs to be stored is bounded by $\frac{1}{2}(m+1)t^2$. If t is small and $m < n$, that could easily be less than the $\frac{1}{2}n^2$ required for the direct Hessian approximation. Typically, for n large, $t^2 \ll n$. Consequently, the storage required for all the Hessians is similar to that required for the sparse Jacobian matrix.

It is possible for the number of nonlinear variables in the Hessian of the Lagrangian $\nabla^2_x \mathcal{L}(x, \lambda)$ to be large even when t is small, and $\nabla^2_x \mathcal{L}(x, \lambda)$ could have little or no sparse structure. While it is necessary to store each component Hessian, storing the full approximation matrix B_k can be avoided by using the objective and constraint Hessians for matrix-vector products with B_k as needed. By contrast, a direct matrix update cannot automatically take advantage of the sparse structure of the problem. Even if the initial approximation matrix is sparse, fill-in is not easily prevented by the BFGS update.

But the assumption that t is small fails even if there is just one constraint with $O(n)$ nonlinear variables. Though the other constraints may be sparse, the storage requirement goes up to $\frac{1}{2}n^2 + O(m)$, which may be prohibitive for large-scale problems. For the case where t is large, the disaggregated model can be implemented using a limited-memory approach to approximate some or all of the component Hessians. For each function with t nonlinear variables, this would require storage of $L(t+1)$ elements, where L is the number of limited-memory updates stored. Such an approach is recommended only if $t > 2L(t+1)/t \approx 2L$. The total storage requirement all component Hessians would add up to $L(t+1)(m+1)$ elements.

4.8 Analysis of the disaggregation model

Our analysis of the disaggregated SR1-based Hessian approximation examines first the consequences for convergence rate and second the memory requirements and computation time involved in this method.

4.8.1 Accuracy and convergence rate

As mentioned earlier, the convergence rate is highly dependent on the accuracy of the Hessian approximation. Several components of the disaggregation model contribute to Hessian accuracy:

- Because we allow an indefinite approximation, the update can be performed at more iterations. Skipping updates leads to poor performance of the SQP algorithm.
- Because the approximation is independent of the multiplier estimates, the Hessian approximations can converge even while the multiplier estimates are rapidly changing.
- Because the estimate is disaggregated, exact second derivatives can be easily incorporated.

For quadratic functions, the disaggregated SR1 update will satisfy the requirement (1.9) for superlinear convergence after a bounded number of iterations. No such claim can be made for the direct BFGS update.

Suppose the objective function and all constraints are quadratic functions. Then, after n linearly independent search directions we could estimate the constraint Hessians exactly. This follows directly from Theorem 4.3.1. Consequently, the sufficient condition on the Hessian approximation for superlinear convergence is satisfied after n iterations.

Still supposing the functions are all quadratics, in some cases the Hessian approximation can be proven to converge in fewer than n iterations. Let S_i be the subspace corresponding to the nonlinear variables for each constraint i, and S_0 for the objective function. For example, if the ith constraint is a function of x_2 and x_3, $c_i(x_2, x_3)$, then $S_i = \mathbb{R}^2(x_2, x_3)$. Suppose there are at most t nonlinear variables for any constraint or objective function, and that the individual Hessians are all nonsingular. Then the Hessian approximations converge as soon as for each constraint i, the search directions projected into S_i span S_i (we cannot require linear independence because they will surely be linearly dependent if the dimension of S_i is less than t). This can occur in as few as t iterations. It follows that the sufficient condition for superlinear convergence could be satisfied in as few as t iterations.

4.8.2 Resources: storage and computation time

A naive implementation of the disaggregation approach for the dense case requires significantly more memory to store the Hessian approximation than the direct approach. The disaggregation approach may require more *computation time* because m quasi-Newton updates could involve more work than a single direct update.

But for the large-scale case where the constraint and objective Hessians are sparse, the approach described in Section 4.7 allows us to compute the Hessian approximation with *greater accuracy* and *lower storage* requirements than the direct method. Moreover, in the sparse case the total computation time also could be less than for the direct method, if a complete quasi-Newton update is computed.

For the large-scale case (limited-memory or sparse) it can take up to m times longer to compute the matrix-vector product $B_k x$. However, many of these more time consuming operations can take advantage of parallel computer architectures. Hessian updates can be computed in parallel, as can the matrix-vector product in some cases. Owing to greater Hessian accuracy we expect the disaggregated method to converge in fewer major iterations. There is a tradeoff, but in certain cases the disaggregated approach promises to be more efficient.

4.9 Numerical example

The following quadratically constrained quadratic program (QCQP) illustrates

1. the problem with changing multiplier estimates,

2. the advantages of disaggregation over aggregation, and

3. the advantages of SR1 over BFGS.

The problem has a relatively simple form, but if the aggregated BFGS Hessian approximation cannot succeed on this type of problem, there is little hope it can be shown to perform well on general problems.

For three quasi-Newton Hessian approximation strategies and for several combinations of initializations for x, the multiplier estimate π, and the Hessian approximation matrix B, we applied the SQP method to the problem given below and compared their performance. Tests on this problem showed that the disaggregated SR1 Hessian gave a more accurate approximation than either the aggregated (direct) or the disaggregated BFGS update. Moreover, the SQP method converged most rapidly using the SR1 Hessian approximation.

4.9.1 Test problem

Given a fixed, positive-definite $H \in I\!R^{5\times 5}$, consider the problem

$$\begin{array}{rl} \underset{x\in I\!R^5}{\text{minimize}} & \frac{1}{2}x^T Hx - e^T x \\ \text{s.t.} & \frac{1}{2}(x^T x - 1) = 0, \end{array} \tag{4.12}$$

where e denotes the vector of all ones. This test problem is small (five variables) and relatively easy. It has a strictly convex quadratic objective and a single quadratic equality constraint, and hence a nonconvex feasible region.

For problem (4.12) let

$$H = \begin{pmatrix} 0.026 & 0 & 0 & 0 & 0 \\ 0 & 0.92 & 0 & 0 & 0 \\ 0 & 0 & 0.7 & 0 & 0 \\ 0 & 0 & 0 & 0.19 & 0 \\ 0 & 0 & 0 & 0 & 0.87 \end{pmatrix}. \tag{4.13}$$

Given H, the solution (to four digits precision) has the optimal Lagrange multiplier $\pi^* = -1.7869$ and the optimal primal variable

$$x^* = \begin{pmatrix} 0.5516 \\ 0.3694 \\ 0.4021 \\ 0.5059 \\ 0.3764 \end{pmatrix}. \tag{4.14}$$

Within the SQP method, the QP subproblem at $x = \bar{x}$ is

$$\underset{p \in I\!R^5}{\text{minimize}} \quad g^T p + \tfrac{1}{2} p^T B p \tag{4.15}$$

$$\text{s.t.} \quad Ap = -\tfrac{1}{2}(\bar{x}^T \bar{x} - 1), \tag{4.16}$$

where $g = H\bar{x} - e$, $A = \bar{x}^T$, and B is an approximation for the Hessian of the Lagrangian:

$$B \approx \nabla^2_{xx}\mathcal{L}(x, \pi) \quad = \quad H - \pi I.$$

4.9.2 Test method

We tested a simplified SQP method using three different Hessian approximation strategies and two ways of initializing the Hessian approximation, for a total of 6 combinations. The three Hessian approximation strategies were

1. Direct, aggregated BFGS.
2. Disaggregated BFGS.
3. Disaggregated SR1.

The two ways of initializing the Hessian approximations were :

1. **Exact** (complete information). Given a Lagrange multiplier estimate π_0, the initial approximation to the Hessian of the Lagrangian is $B_0 = \nabla^2_{xx}\mathcal{L}(x_0, \pi_0)$.
2. **General** (zero information). Initialize with identity matrix, i.e., $B_0 = I$.

At first we tried three different initial point estimates for x and λ:

1. an arbitrary point
2. a feasible point
3. a point close to the solution

However, for the cases run, the results obtained were very similar.

The SQP method was applied as follows. Since the example is small, the QP subproblems were solved directly from the KKT system to get the search direction and multiplier estimate. For simplicity, there was no linesearch, and the full step of size one was always taken. The multiplier estimates were updated by setting them to be the QP multipliers, and only the most recent multiplier estimates were used in the quasi-Newton update. For this test problem the exact Hessian of the Lagrangian is positive definite at all x including x^* and for all $\pi < 0$. The initial starting points were selected so the Hessian approximation update would usually not need to be skipped (except when close to the solution). No modifications were made but Powell's rule for BFGS and corresponding rules for SR1 were used to decide when to skip the update.

With the SQP method implemented as above, the Hessian approximation methods were compared according to the convergence rate of the algorithm and the error in the Hessian approximation. The convergence rate is measured by the rate of decrease in the norm of the gradient of the Lagrangian. Note that the convergence of the norm of the infeasibilities to zero is independent of the Hessian approximation. Generally, the norm of the constraint violations converges to zero faster than the norm of the gradient of the Lagrangian. The *relative* error in the Hessian approximation (that is, compared to the Hessian of the Lagrangian using exact second derivatives and the current QP multiplier estimates) is measured by

$$\frac{\|(B_k - \nabla^2 \mathcal{L}(x_k, \pi_k)\|}{\|\nabla^2 \mathcal{L}(x_k, \pi_k)\|}. \tag{4.17}$$

4.9.3 Performance results

The comparative performance of a simple SQP routine with different quasi-Newton approximations is summarized in Figures 4.1 and 4.2. In the first figure the Hessian approximation is initialized to the identity matrix ($B_0 = I$). In the second figure the Hessian approximation is initialized using exact second derivatives, i.e., $B_0 = \nabla^2_{xx}\mathcal{L}(x_0, \pi_0)$.

For each figure, there are two graphs:

1. on the left, the log of the norm of the gradient of the Lagrangian versus the number of iterations, and

2. on the right, the "relative error" in the Hessian approximation (4.17) versus the number of iterations.

Note that both performance measures are graphed in log plots to allow a large range of values to be depicted.

The relative error of the Hessian approximation was much smaller for the disaggregated

SR1 than for the aggregated BFGS. Several observations about the Hessian approximation for the QCQP might explain why:

- If an aggregated Hessian approximation is used, it is possible to start with an approximation based on the exact second derivatives but inexact multiplier estimates, and have the entire approximation rapidly deteriorate as the multiplier estimates change.

- If the disaggregated SR1 Hessian approximations (components updated using either SR1 or BFGS) are initialized using the exact Hessian at the initial point (x, π), the Hessian approximations do not deteriorate. Rather, the exact Hessian of the Lagrangian (with updated multiplier estimates) is produced at all iterations. The SR1 update is actually skipped if the quasi-Newton condition is already satisfied. The BFGS update is not skipped; however, the two terms of the BFGS update cancel each other out (subject to numerical error) if the quasi-Newton condition, $y = B\delta$, is already satisfied (which it is for quadratic functions as long as the BFGS update is not modified for positive definiteness). Note that the approximations will satisfy the quasi-Newton condition at all iterations because the functions are quadratic and hence the true Hessians are constant.

- Even if not initialized with exact second derivatives, the SR1 disaggregated approach is proven to converge in fewer iterations than the number of variables, if the search directions are independent (see Theorem 4.3.1).

It is another matter whether the SQP method itself will converge much faster with the better Hessian approximation. According to the sufficient condition, the Hessian approximation need not be exact in order for the algorithm to converge superlinearly.

With the direct BFGS update, the gradient of the Lagrangian converged to zero linearly (with a small constant) for each of the three starting points. Using the disaggregated SR1 update, superlinear convergence, sometimes even quadratic, was observed.

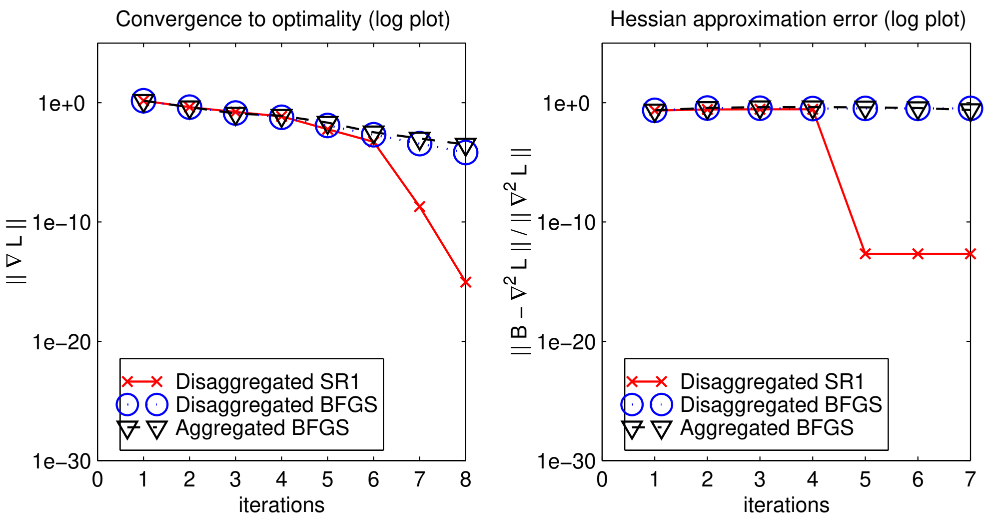

Figure 4.1: Comparative performance with Hessian approximation initialized to the identity matrix.

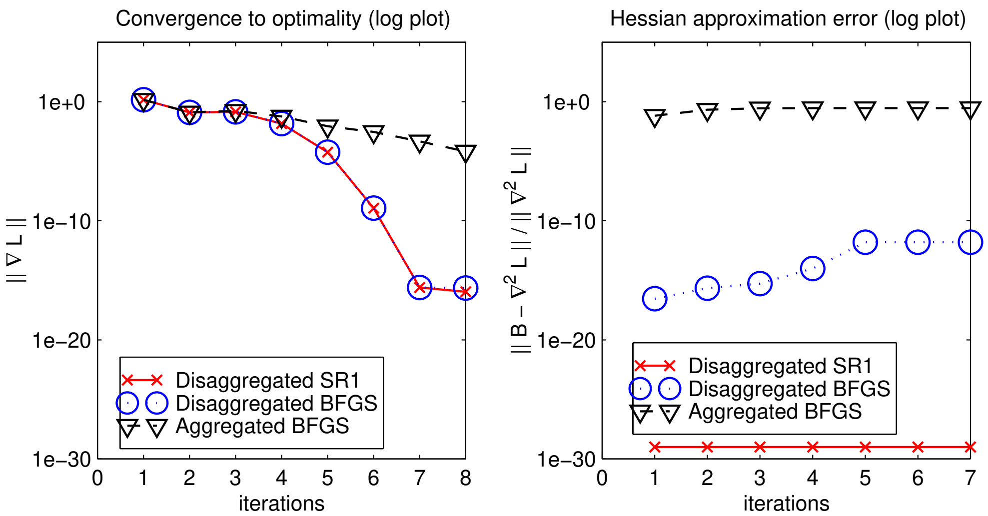

Figure 4.2: Comparative performance with Hessian approximation initialized using exact second derivatives.

Chapter 5

Nonconvex QP Subproblems

5.1 Introduction

The use of the disaggregated, SR1 quasi-Newton approximation to the Hessian of the Lagrangian leads to approximations, and hence QP subproblems, that are indefinite. The SQP algorithm PDSQP described in Chapter 2 will not work with such Hessian approximations, but IDSQP does allow nonconvex subproblems. The null-space active-set method based subroutine IDQP that finds a descent direction for IDSQP has certain drawbacks, namely that in order to ensure convergence of IDSQP it always terminates at the first constrained stationary point, and that the descent step off the stationary point is based on the steepest-descent direction.

The purpose of this chapter is to explore alternative ways to compute the descent direction. We also describe how to compute the initial modified reduced Hessian, which is needed for an implementation of IDQP. We present an alternative algorithm and show that the descent direction produced satisfies requirements for the outer SQP algorithm IDSQP to be convergent. Requirements for convergence are based on the algorithms described in Chapter 2.

The options explored include modifying the active-set method by altering the QP Hessian and/or terminating the procedure before reaching a minimizer. While not a requirement, a strong preference is that the modification be minimal, in the sense that no change should be made to the original method for solving the QP if the solution to the unmodified problem is satisfactory. For example, when the reduced Hessian is positive definite at a solution, we do not want to alter the subproblem in the neighborhood of the solution, even though the Hessian may be indefinite. The modified Newton method for unconstrained optimization follows similar principles and was the basis for our approach to constrained optimization with indefinite subproblems.

We begin by describing the issues involved in generalizing the modified Newton approach and outlining the range of options for modifying the QP.

5.1.1 Generalizing the modified Newton approach

Modified Newton methods for unconstrained optimization

Newton methods for unconstrained optimization take steps that minimize local quadratic models of the function based on the exact Hessians at each iterate. A requirement that the quadratic model be positive definite ensures a minimum exists, and the minimizer is a search direction of sufficient descent. The "modified Newton" approach attempts to use the exact Hessian even when it is not positive definite. To ensure the subproblem has a minimizer, this approach sets the model Hessian equal to a positive-definite "approximation" of the true Hessian.

The modified Newton procedure follows two principles:

1. The algorithm is required to produce a descent direction.

2. The modification should be minimal; no change should be made to the model Hessian when it is positive definite, as it usually is in the neighborhood of the solution.

For unconstrained optimization, the first principle is satisfied as long as the modified Hessian is sufficiently positive definite. Unfortunately, it is not immediately obvious whether a given matrix is positive definite. To increase the likelihood that the matrix is positive definite, one could add a large multiple of the identity matrix, but that would not be a minimal change, violating the second principle.

Cholesky factorization A simple way to make a minimal change to the Hessian, without unnecessary computational expense, is to use the Cholesky factorization both to solve the system of equations for the search direction and to modify the Hessian. For background on the Cholesky factorization, see [GV89] or [GMW81, pp. 36–37]. The Cholesky factorization only exists for positive-definite matrices, but if in the course of performing the factorization it becomes apparent that the matrix is not positive definite, at that point the matrix can be modified to enable the algorithm to continue. It may be necessary to make a series of changes to the original matrix. On completion, we have Cholesky factors for some matrix that is obviously positive definite. A nice feature of the methods to find Cholesky factors is that the work required is little more than if the final matrix were known initially and the pure Cholesky algorithm used. For more details about the modified Cholesky factorization, see [GMW81, pp. 108–111].

Modified Newton methods for constrained problems

Consider now the constrained case where the local model is a quadratic program and the Hessian of the QP is indefinite. The same principle applies to the desired modification: while providing a descent direction satisfactory for a convergent SQP algorithm, the change should also be minimal, with a computational cost similar to the convex case.

As in the unconstrained case, if the full Hessian H is not positive definite, the QP may not have a solution, or when a minimizer exists, it may not be a descent direction from p_0 for the QP or the merit function. On the other hand, for constrained problems the important curvature is of the reduced Hessian rather than the full Hessian. In contrast to the full Hessian (a large matrix not assumed to be positive definite even at the solution), the reduced Hessian is a smaller matrix that can be assumed positive definite at the solution. If the initial reduced Hessian is modified to be positive definite, a first constrained stationary point will exist. We do not want to modify the full Hessian because that would not be minimal, would be expensive, and could impede the rate of convergence. But, modifying the Cholesky factorization of the reduced Hessians while solving the QP would not require much additional work. If the Hessian could be modified initially, in a minimal way, to be positive definite when projected into all subspaces encountered and the union of all subspaces encountered, then the result would be a descent direction for the merit function. Unfortunately, if the active set at the initial point is not the active set at the solution, it is difficult to ensure the reduced Hessian will remain positive definite on all subspaces encountered.

5.1.2 Options for modifying the QP

Five possible ways to modify the QP subproblem follow.

(i) Alter the full Hessian to be positive definite. For example, for some $\gamma > 0$, let

$$\bar{H} \leftarrow H + \gamma I,$$

then solve the positive definite QP. One way to determine λ so that $\bar{H}$ is positive definite would be to perform a spectral decomposition and set λ to be greater in magnitude than the smallest negative eigenvalue. To avoid computing the eigenvalues (an expensive operation), Betts *et al.* [BH93] applied the Gershgorin theorem [GV89, pp. 341–2] to determine a lower bound on the smallest eigenvalue.

A significant drawback to this approach is that it may alter even a well-conditioned, positive-definite matrix. Since the Hessian is not positive definite in the neighborhood of a solution, this approach is likely to cause a serious deterioration in the rate of convergence. The next three approaches aim to avoid unnecessary modification.

(ii) To avoid modifying matrices unnecessarily, we could first determine whether the initial reduced Hessian is positive definite (see Section 5.2). If not, we could then apply (i). Unfortunately, the fact that the initial reduced Hessian is positive definite does not ensure a satisfactory solution exists to the QP. Later reduced Hessians may still be indefinite.

(iii) Murray and Prieto's approach, described in Chapter 2, alters the QP Hessian at the initial feasible point so the initial reduced Hessian is positive definite, ensuring the existence of the first constrained stationary point. Their algorithm finds the constrained stationary point using an active-set method and computes a descent direction based on Lagrange multiplier estimates at the stationary point. Continuing to a later stationary point is allowed as long as the modified QP Hessian is positive definite on the subspace spanned by all search directions encountered, but it is difficult to ensure or even check that this condition holds without stopping at the first stationary point.

(iv) Start by following (iii), then move past the first stationary point. Attempt to solve the QP using a null-space active-set method. At each iteration, alter the QP as necessary to ensure the reduced Hessian is positive definite on every subspace encountered. Then show the solution obtained is the solution of a related QP that is the same except for having a positive-definite Hessian.

Unfortunately, there is a counterexample (provided in Section 5.3.2) that shows such a strictly convex problem may not exist even if the reduced Hessian is positive definite on every subspace. If no such strictly convex problem exists, the solution may not be a sufficient descent direction for the merit function.

(v) Follow (iv), but stop if necessary in order to guarantee a sufficient descent direction and satisfy requirements for the SQP method to converge.

This last approach does not alter the problem in the neighborhood of the solution, yet it produces a descent direction for the merit function.

The above approaches to dealing with a nonconvex QP focus on modifying the approximation to the Hessian of the Lagrangian. A different approach, which we have not considered, would have been to add constraints or bounds to the QP instead of modifying the Hessian.

5.1.3 Preview

The remainder of this chapter is organized as follows.

- In Section 5.2 we describe modifications to the initial reduced Hessian that satisfy the requirements of the indefinite SQP algorithm IDSQP described in Chapter 2.

- In Section 5.3 we explore how one might proceed with the null-space active-set method beyond the first stationary point. We provide a cautionary counterexample to show the need for early termination conditions even when no changes need be made to the QP Hessian.

- In some cases, the termination conditions allow modifying the Hessian additional times at or after the first constrained stationary point in order to maintain a positive-definite reduced Hessian. This allows the algorithm to continue towards a minimizer. Implementation details for such modifications are given for the dense and sparse cases in Section 5.4. After analyzing properties of the modification procedure, we show how a descent step off the final stationary point can be constructed by a natural continuation of the algorithm past the final stationary point.

- In Section 5.5, we present a detailed statement of the algorithm with pseudocode.

- It is shown that the algorithm terminates in Section 5.6.1.

5.2 Initial modifications to the QP Hessian $H^{(0)}$

We describe how to modify the full Hessian and the reduced Hessian initially in both the dense and sparse cases, in order to produce a positive-definite reduced Hessian. We use the notation introduced in Sections 2.1.3 and 2.2.1.

The initial reduced Hessian may require extensive modification, but since changes to the objective do not affect the choice of initial feasible point and can be assumed to have been made before we start to solve the subproblem, we have a lot of freedom in choosing this modification. A variety of initial modifications to the reduced Hessian have been proposed [FGM95, GM74]. While the spectral decomposition would produce a minimal modification with nice properties, it is inefficient and impractical to compute for large problems. We wish the modification to have certain properties. The modification should be minimal and economical to compute, $H^{(0)}$ must be bounded, and the reduced Hessian needs to have positive, bounded eigenvalues.

One approach is based on the partial Cholesky factorization [FGM95]. For clarity of exposition we omit the symmetric permutation used by the partial Cholesky algorithm.

The set of equations to be solved at the initial point is $Z_0^T H_0 Z_0 d = -Z_0^T g_0$. If $Z_0^T H_0 Z_0 \succ 0$, this system can be solved using a full Cholesky factorization. Even if $Z_0^T H_0 Z_0 \not\succ 0$, the Cholesky factorization (with permutations) can proceed until there are no remaining positive diagonals or no positive diagonals greater than some $\beta_{HZ} > 0$:

$$Z_0^T H_0 Z_0 = \begin{pmatrix} L_1 & \\ L_2 & I \end{pmatrix} \begin{pmatrix} I & \\ & S \end{pmatrix} \begin{pmatrix} L_1^T & L_2^T \\ & I \end{pmatrix}, \tag{5.1}$$

where L_1 is a lower-triangular matrix with a bounded condition number. This is called the "partial Cholesky factorization." The modified reduced Hessian that results from this factorization is

$$Z_0^T H_0^{(0)} Z_0 = \begin{pmatrix} L_1 & \\ L_2 & I \end{pmatrix} \begin{pmatrix} L_1^T & L_2^T \\ & I \end{pmatrix}. \tag{5.2}$$

Since the resulting modified Hessian is bounded, as long as H_0 is bounded, the size of the modification is also bounded.

There are other bounded options for modifying the reduced Hessian, such as the "modified Cholesky factorization" [GM74]. For the current discussion we simply assume T is known such that $\|T\| < \infty$ and

$$Z_0^T H_0^{(0)} Z_0 = Z_0^T H_0 Z_0 + T \tag{5.3}$$

is sufficiently positive definite. Note that T is an $n_Z \times n_Z$ matrix, where n_Z is the number of columns in Z_0.

The question is, what does this imply about $H^{(0)}$? This matrix is needed to compute the gradient $\nabla\psi^{(0)}(p)$. Let M be a matrix such that

$$H^{(0)} = H + M, \tag{5.4}$$

subject to (5.3) also holding. It is obvious that M is not uniquely defined by (5.3) and (5.4).

5.2.1 Dense case

Suppose Z_0 has orthonormal columns, so $Z_0^T Z_0 = I$. Then

$$H_0^{(0)} \equiv H_0 + Z_0 T Z_0^T, \tag{5.5}$$

clearly satisfies (5.3). From the definition of $\bar{P}_0$ in (2.7),

$$\bar{P}_0^T H^{(0)} \bar{P}_0 \equiv \bar{P}_0^T H \bar{P}_0 + Z_0 T Z_0^T \tag{5.6}$$

is equivalent to (5.5). It follows that

$$H^{(0)} = H + \bar{P}_0 (Z_0 T Z_0^T) \bar{P}_0^T.$$

When Z_j has orthonormal columns, $H^{(0)}$ can be computed simply by defining

$$M \equiv \bar{P}_0 (Z_0 T Z_0^T) \bar{P}_0^T. \tag{5.7}$$

This is the choice of M satisfying (5.3) that minimizes the Frobenius norm of M.

5.2.2 Sparse case

For large, sparse problems, it is inefficient for Z_0 to have orthonormal columns, but $H^{(0)}$ can be determined as follows. Start by expanding the relationship between T and M. Multiply equation (5.4) by $\bar{P}_0^T$ on the left and $\bar{P}_0$ on the right, yielding

$$\bar{P}_0^T H^{(0)} \bar{P}_0 = \bar{P}_0^T H \bar{P}_0 + \bar{P}_0^T M \bar{P}_0.$$

It follows that

$$H_0^{(0)} = H_0 + \bar{P}_0^T M \bar{P}_0. \tag{5.8}$$

Given the definition of T from (5.3), multiplying (5.8) by Z_0^T on the left and Z_0 on the right implies

$$Z_0^T (\bar{P}_0^T M \bar{P}_0) Z_0 = T. \tag{5.9}$$

For sparse problems a common form for Z_0 is

$$Z_0 = \begin{pmatrix} W \\ I_{n_Z} \end{pmatrix},$$

where I_{n_Z} is the identity matrix with dimension equal to the number of columns in Z (that is, $n -$ rank(A_0)). It follows that a convenient choice for M satisfying (5.9) is given by

$$M \equiv \bar{P}_0 \bar{T} \bar{P}_0^T, \tag{5.10}$$

where

$$\bar{T} = \begin{pmatrix} 0 & 0 \\ 0 & T \end{pmatrix}.$$

5.3 Moving beyond the first constrained stationary point

The subroutine IDQP to find a descent direction, described in Chapter 2, stops at the first constrained stationary point. We wish to have the option of moving beyond the first constrained stationary point and perhaps finding a minimizer. We would like to follow the positive-definite QP subroutine PDQP described in Chapter 2 as closely as possible, but as explained earlier this procedure might fail because the reduced Hessian could become indefinite, and even if it did not fail, other conditions for convergence might not hold.

5.3.1 Further modifications to the Hessian of the Lagrangian

It is possible to extend IDQP past the first stationary point, in such a way that at least it does not fail immediately. We repeatedly change the QP problem being solved, at the initial point replacing H by $H^{(0)}$ and subsequently replacing $H^{(j)}$ by $H^{(j+1)}$ to ensure a positive-definite reduced Hessian. The objective minimized at the jth QP iteration would be

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

The choice of $H^{(j+1)}$ would depend on whether $Z_{j+1}^T H_j^{(j)} Z_{j+1} \succ 0$. If so, then $H^{(j+1)} = H^{(j)}$. If not, $H^{(j+1)}$ is chosen such that $Z_{j+1}^T H_j^{(j+1)} Z_{j+1} \succ 0$. Later in this chapter, we discuss algorithms for modifying the matrix as necessary. Since earlier constrained stationary points may no longer be stationary points after the Hessian is modified, it needs to be proven that the algorithm terminates if the Hessian is modified more than once.

Recall that the QP subproblem being solved is of the form BQP (1.17), having equality constraints and lower bounds on the variables. When an additional variable moves onto its bound, the reduced Hessian remains positive definite. The only occasion on which a reduced Hessian is encountered that is not positive definite is after a stationary point is reached, when a variable is released from its bound (increasing by one the number of free variables). In this case, Z_j differs from Z_{j+1} by the addition of an extra row and column. (In general, we need only consider modifying the Hessian at the initial feasible point and immediately after a bound is deleted from the working set; in all other cases the Hessian remains the same.)

Desirable Properties

We desire the following properties to hold for modifications to the Hessian and reduced Hessian.

1. **Low rank change to H.** The change from $H^{(j)}$ to $H^{(j+1)}$ should be of low rank (e.g., rank one).

2. **Positivity of reduced Hessian.** If the algorithm terminates after J iterations, we would like $Z_j^T H^{(J)} Z_j \succ 0$ for $j = 0, \ldots, J$.

3. **Condition of reduced Hessian.** The minimum eigenvalue of the reduced Hessian $Z_j^T H_j^{(j)} Z_j$ should be positive, bounded away from zero, and bounded above. Provided a finite number of modifications are made, it would follow that $H^{(j)}$ is bounded above.

4. **Constancy of reduced Hessian.** The previous reduced Hessian should remain unchanged, i.e., $Z_j^T H_j^{(j+1)} Z_j = Z_j^T H_j^{(j)} Z_j$.

5. **Stationary point.** After $H_j^{(j)}$ is replaced by $H_j^{(j+1)}$, it is desirable that p_j remain a constrained stationary point of the new problem. Otherwise, at the next iteration we will continue moving on the same subspace, regardless of how the multipliers change.

6. **Multipliers.** It is desirable for the multipliers to remain the same, or at least for the sign of the multiplier corresponding to the deleted bound to remain negative, so that we move off the bound at the next iteration.

5.3.2 Cautionary example

Unfortunately, even if all the above objectives are achieved, this approach cannot ensure the final point is an adequate search direction. The following example shows that even if the reduced Hessian remains positive definite on every subspace without modification and a local minimizer is reached, the step to the minimizer may not be a descent direction for the merit function.

Consider the QP

$$\boxed{\begin{array}{ll} \underset{p\in I\!R^3}{\text{minimize}} & g^Tp + \frac{1}{2}p^THp \\ \text{subject to} & \begin{pmatrix} -\infty \\ 0 \\ 0 \end{pmatrix} \le p \le \begin{pmatrix} .5 \\ 1 \\ 1.5 \end{pmatrix}, \end{array}}$$

where

$$H = \begin{pmatrix} 1 & .5 & -.5 \\ .5 & 1 & -2.5 \\ -.5 & -2.5 & 1 \end{pmatrix} \qquad \text{and} \qquad g = \begin{pmatrix} -1 \\ -1.25 \\ 1.25 \end{pmatrix}.$$

When one applies PDQP starting from the initial feasible point of zero, on every subspace up to the minimizer the reduced Hessian is positive definite, so no modifications appear to be needed. The minimizer p^* is

$$p^* = \begin{pmatrix} .5 \\ 1 \\ 1.5 \end{pmatrix}.$$

While the objective function is lower than at the initial point, we have $g^T p^* = 0.125 > 0$ and p^* cannot be a descent direction for the merit function for any value of ρ. The fundamental problem is that p^* is a direction of negative curvature ($p^{*T} H p^* = -4.25$).

Also note that there is *no* positive-definite H for which p^* would be a solution of the QP, given the same g. This shows you cannot just solve an indefinite QP for p^* and afterwards expect to find a positive-definite QP for which p^* is a descent direction.

5.3.3 Theoretical considerations

The proof of convergence for Murray and Prieto's second-derivative algorithm SQP2D allows moves to other than the first stationary point, but only under a stronger positivity assumption on the Hessian of the modified QP than that the initial reduced Hessian be positive definite (see Section 2.2.1). Moreover, it is assumed the Hessian is modified at most once, at the initial feasible point. Unfortunately, the positivity assumption is difficult to check and unlikely to hold true in all cases.

We need to eliminate our reliance on the positivity assumption in order to construct a practical algorithm that can move beyond the first stationary point. Let us examine how the positivity condition of Section 2.2.1 is used in the proofs of Lemmas 3.4 and 3.5 in [MP99], upon which Murray and Prieto's theorems asserting global convergence of the SQP algorithm, at a superlinear rate and with a bounded penalty parameter, rely.

Lemma 3.4

Lemma 3.4 of [MP99] relies on the positivity assumption to show positive curvature in the total direction from the initial feasible point p_0, that is,

$$(\tilde{p} - p_0)^T \tilde{H} (\tilde{p} - p_0) \geq \beta_{HZ} \|\tilde{p} - p_0\|^2, \tag{5.11}$$

at the final stationary point $\tilde{p}$ reached by the QP algorithm. (The lemma also shows that the modified Hessians $\tilde{H}$ over the set of SQP iterations k are bounded, but assuming the Lagrange multiplier estimates are bounded, this follows from the way we construct the Hessian approximations and the Hessian modifications within the QP. Another portion of the lemma deals with directions of negative curvature, but that is irrelevant because we are ignoring directions of negative curvature.)

Even without the general positivity assumption, condition (5.11) holds for the very *first* stationary point $\tilde{p}$, by the following argument. Suppose N steps were taken from the initial feasible point to the first stationary point $\tilde{p} = p_N$. The value of $\tilde{p}$ can be written as a sum,

$$\tilde{p} = p_0 + \sum_{j=0}^{N-1} \gamma_j \bar{d}_j,$$

where $\bar{d}_j$ denotes the QP search direction in the jth QP iteration and $\gamma_j \leq 1$ the step taken along $\bar{d}_j$. Let p_ℓ denote the ℓ-th partial sum, $p_\ell = p_0 + \sum_{j=0}^{\ell-1} \gamma_r \bar{d}_r$, associated with the ℓ-th QP iteration. Since bounds have only been added, not deleted, the null-space matrix has become progressively smaller, allowing every $\bar{d}_j$ to be written as $Z_0 \bar{d}_j^Z$, where Z_0 is the null-space matrix at p_0.

Thus, we can write $(\tilde{p} - p_0)$ as a vector in the initial null-space:

$$(\tilde{p} - p_0) = \sum_{j=0}^{N-1} \gamma_j \bar{d}_j = Z_0 \sum_{j=0}^{N-1} \gamma_j \bar{d}_j^Z,$$

and

$$\begin{aligned} (\tilde{p} - p_0)^T \tilde{H} (\tilde{p} - p_0) &= (\sum_{j=0}^{N-1} \gamma_j \bar{d}_j^Z)^T Z_0^T \tilde{H} Z_0 (\sum_{j=0}^{N-1} \gamma_j \bar{d}_j^Z) \\ &> \beta_{HZ} \| \sum_{j=0}^{N-1} \gamma_j \bar{d}_j^Z \| \end{aligned} \tag{5.12}$$

if $Z_0^T \tilde{H} Z_0 \succ 0$. This satisfies (5.11).

If $\tilde{p}$ is not the first stationary point, instead of checking the general positivity assumption (too difficult!) we can verify (5.11) directly.

Lemma 3.5

The positivity assumption is also used in [MP99] to prove Lemma 3.5, which states that there exists a constant $\beta_{gp} > 0$ such that at the stationary point $\tilde{p}$,

$$g^T\tilde{p} \leq -\frac{1}{2}(\tilde{p}-p_0)^T\tilde{H}(\tilde{p}-p_0) + \beta_{gp}\|\tilde{c}^-\|. \tag{5.13}$$

Murray and Prieto's proof of Lemma 3.5 (5.13) cites the "positive definiteness of $\tilde{H}$ on the relevant subspaces, and the convexity of $\phi(y) = y^T\tilde{H}y$ on those subspaces." The main purpose of citing convexity is to justify the bound

$$N\sum_{j=0}^{N-1}\gamma_j^2\bar{d}_j^T\tilde{H}\bar{d}_j \geq (\tilde{p}-p_0)^T\tilde{H}(\tilde{p}-p_0). \tag{5.14}$$

Their proof also relies upon having $(g+\tilde{H}p_j)^T\bar{d}_j < 0$ for all j. By the definition of the QP step $\bar{d}_j$, this holds as long as $\tilde{H} = H^{(0)}$.

Thus, as long as the Hessian has not been modified after the initial feasible point, the positivity assumption can be replaced by checking (5.11) (which is needed anyway) and (5.14). But, suppose at a stationary point $\tilde{p}$, $\tilde{H}$ *has* been modified at least once after the first stationary point (i.e., $\tilde{H} \neq H^{(0)}$). Even if we could determine at each step whether (5.14) holds, the proof of Lemma 3.5 [MP99] is invalidated, because it is no longer necessarily true that $(g+\tilde{H}p_j)^T\bar{d}_j < 0$. The lemma might still hold, however, as can be determined by testing (5.13) directly.

5.3.4 Conditions for satisfactory stationary point

The counterexample showed that even if we reach a minimizer of some QP, the fact that the problem is still nonconvex leaves open the possibility that it may *not* be a descent direction, i.e.,

$$(g+\bar{H}p_0)^T(p^*-p_0) > 0. \tag{5.15}$$

If the Hessian is modified during the algorithm, the objective value is altered at both the current point and the initial feasible point. Since we always increase the objective at both points, it is possible for the final point to be worse than the initial point, i.e.,

$$\tilde{\psi}(p^*) > \tilde{\psi}(p_0). \tag{5.16}$$

Making a more extensive change from H to $H^{(0)}$ would reduce the likelihood of needing further changes, but would not eliminate the difficulty altogether.

The convergence results proved in [MP99] hold as long as the following three conditions hold at the terminal stationary point $\tilde{p}$. The first condition requires the QP objective value to be decreasing. The second condition is the directional positive curvature (5.11). The third condition is (5.13) or (5.14), needed to prove Lemma 3.5 of [MP99].

(i)
$$\tilde{\psi}(\tilde{p}) < \tilde{\psi}(p_0). \tag{5.17}$$

(ii)
$$(\tilde{p} - p_0)^T \tilde{H}(\tilde{p} - p_0) > \beta_{HZ}\|\tilde{p} - p_0\|^2. \tag{5.18}$$

(iii)
$$(\tilde{p} - p_0)^T \tilde{H}(\tilde{p} - p_0) \leq 2(\beta_{gp}\|\tilde{c}^-\| - g^T\tilde{p}), \tag{5.19}$$

where $\tilde{c}_i$ denotes the normalized constraints $\tilde{c}_i = c_i/(1 + \|a_i\|)$ and a_i is the ith row of A. If the Hessian has not been modified since the initial feasible point, the weaker bound
$$(\tilde{p} - p_0)^T H^{(0)}(\tilde{p} - p_0) \leq N \sum_{j=0}^{N-1} \gamma_j^2 \bar{d}_j^T H^{(0)} \bar{d}_j \tag{5.20}$$
will suffice, where N is the number of steps taken to compute $\tilde{p}$ (the QP stationary point).

Remark 5.3.1 *Conditions (5.17) and (5.18) imply that the direction $\tilde{p} - p_0$ is a descent direction for $\tilde{\psi}$ at p_0, i.e., $\nabla\tilde{\psi}(p_0)^T(\tilde{p} - p_0) = (g + \tilde{H}p_0)^T(\tilde{p} - p_0) < 0$.*

Proof.

$$\begin{aligned}
(g + \tilde{H}p_0)^T(\tilde{p} - p_0) &= g^T\tilde{p} - g^T p_0 + p_0^T\tilde{H}(\tilde{p} - p_0) \\
&= g^T\tilde{p} + \frac{1}{2}\tilde{p}^T\tilde{H}\tilde{p} - g^T p_0 - \frac{1}{2}p_0^T\tilde{H}p_0 - \frac{1}{2}\tilde{p}^T\tilde{H}\tilde{p} - \frac{1}{2}p_0^T\tilde{H}p_0 + \tilde{p}^T\tilde{H}p_0 \\
&= \tilde{\psi}(\tilde{p}) - \tilde{\psi}(p_0) - \frac{1}{2}(\tilde{p} - p_0)^T\tilde{H}(\tilde{p} - p_0) \\
&< 0.
\end{aligned}$$

∎

The three conditions (5.17)–(5.19) hold for $\psi^{(0)}$ at the first stationary point. Unless the Hessian is modified, and it need not be modified before a variable is released from its bound, condition (5.17) holds because the QP objective value will not increase. The second two conditions hold because $\phi(y) = y^T(Z_0^T H^{(0)} Z_0)y$ is strictly convex and the directions $p_j - p_0$ and each of the $\bar{d}_j$ are in the subspace defined by Z_0 (i.e., $\bar{d}_j = Z_0 \bar{d}_j^Z$).

After the first stationary point, it is possible that one of the conditions will fail to hold, whether or not the Hessian has been modified more than once. The conditions must be checked at every stationary point reached, and the QP algorithm is terminated once any of the conditions fail. If the QP subroutine returns the last stationary point at which all three conditions held along with a descent step computed from that point, the validity of the SQP convergence results is preserved.

5.4 Implementing more modifications

When a variable is moved off a bound, the dimensions of Z_{j+1} increase by one over the dimensions of Z_j. Let the index τ correspond to the variable being released from its bound. At this point, two quantities need to be computed: the null-space matrix Z_{j+1} and the Hessian in the new set of free variables:

$$H_{j+1}^{(j)} = \begin{pmatrix} H_j^{(j)} & h_\tau \\ h_\tau & h_{\tau\tau} \end{pmatrix},$$

where h_τ is the vector composed of the elements of the τth column of $H^{(j)}$, and $h_{\tau\tau} = H_{\tau\tau}^{(j)}$.

Given Z_{j+1} and $H_{j+1}^{(j)}$, we can compute the Cholesky factorization of the reduced Hessian $Z_{j+1}^T H_{j+1}^{(j)} Z_{j+1}$. Then, if the new reduced Hessian is not sufficiently positive definite, i.e., if $\lambda_{\min}(R^T R = Z_0^T H_0^{(0)} Z_0) \leq \beta_{HZ}$, the new reduced Hessian is modified and the modification reflected back onto the full Hessian. We differentiate the dense case from the sparse case. The motivation for using a different update in the dense case versus the sparse case is to achieve a modification that is smaller in norm, at the expense of being more expensive to store.

5.4.1 Dense case

Null-space representation On small problems, Z_j can be presumed to have orthonormal columns, and the updated null-space matrix takes the form

$$Z_{j+1} = \begin{pmatrix} Z_j & w \\ 0 & \theta \end{pmatrix}, \qquad z \equiv \begin{pmatrix} w \\ \theta \end{pmatrix}, \tag{5.21}$$

where $0 < \theta < 1$, $Z_j^T w = 0$, and $A_{j+1} z = 0$. Note that w cannot be equal to zero. If a is the new column of A_{j+1},

$$0 = A_{j+1} z = \begin{pmatrix} A_j & a \end{pmatrix} \begin{pmatrix} w \\ \theta \end{pmatrix},$$

which implies $A_j w + \theta a = 0$. If we let $w = 0$, then $\theta > 0$ implies $a = 0$. But that is a contradiction, since the columns of A are assumed to be nonzero, so $w \neq 0$.

Reduced Hessian The reduced Hessian in the new set of free variables is

$$Z_{j+1}^T H_{j+1}^{(j)} Z_{j+1} = \begin{pmatrix} Z_j^T H_j^{(j)} Z_j & Z_j^T H_j^{(j)} w + \theta Z_j^T h_\tau \\ w^T H_j^{(j)} Z_j + \theta h_\tau^T Z_j & w^T H_j^{(j)} w + 2\theta w^T h_\tau + \theta^2 h_{\tau\tau} \end{pmatrix}. \tag{5.22}$$

If in the process of updating the Cholesky factors it emerges that $Z_{j+1}^T H_{j+1}^{(j)} Z_{j+1} \succ 0$, then $H^{(j+1)} = H^{(j)}$.

If the reduced Hessian is not sufficiently positive definite, it can be made positive definite in the following manner. We modify the τ^{th} diagonal element of the Hessian by adding $\delta_j^H > 0$ to it:

$$\begin{aligned} H^{(j+1)} &= H^{(j)} + \delta_j^H e_\tau e_\tau^T \\ H_{j+1}^{(j+1)} &= H_{j+1}^{(j)} + \delta_j^H e_{n_Z} e_{n_Z}^T, \end{aligned} \tag{5.23}$$

for some $\delta_j^H > 0$ chosen sufficiently large, where n_Z is the number of rows in Z_{j+1}. Note that this modification leaves $H_j^{(j+1)} = H_j^{(j)}$. It follows from (5.21) and (5.23) that

$$Z_{j+1}^T H_{j+1}^{(j+1)} Z_{j+1} = Z_{j+1}^T H_{j+1}^{(j)} Z_{j+1} + \delta_j^H \theta^2 e_{n_Z} e_{n_Z}^T. \tag{5.24}$$

If we choose δ_j^H sufficiently large, $Z_{j+1}^T H_{j+1}^{(j+1)} Z_{j+1} \succ 0$. The reduced Hessian is positive definite if and only if its Cholesky factors are well-defined. Since the reduced Hessian at iteration j was positive definite, we can assume its factors

$$R_j^T R_j = Z_j^T H_j^{(j)} Z_j$$

were well-defined. When Z_j changes to Z_{j+1}, the Cholesky factor R_j changes to

$$R_{j+1} = \begin{pmatrix} R_j & r \\ 0 & \varrho \end{pmatrix}, \tag{5.25}$$

and

$$R_{j+1}^T R_{j+1} = \begin{pmatrix} R_j^T R_j & R_j^T r \\ r^T R_j & r^T r + \varrho^2 \end{pmatrix} = Z_{j+1}^T H_{j+1}^{(j+1)} Z_{j+1}. \tag{5.26}$$

From (5.22), (5.24), and (5.26), it follows that the vector r of R_{j+1} is uniquely defined as the solution to

$$R_j^T r = Z_j^T H_j^{(j)} w + \theta Z_j^T h_\tau. \tag{5.27}$$

For the new reduced Hessian to be positive definite, the last diagonal element ϱ of the Cholesky factor R_{j+1} must be positive and bounded away from zero. By (5.26),

$$r^T r + \varrho^2 = w^T H_j^{(j)} w + 2\theta w^T h_j + \theta^2 h_{\tau\tau} + \theta^2 \delta_j^H.$$

Finally, $\varrho > \epsilon$ as long as

$$\delta_j^H > \frac{\epsilon^2 - (w^T H_j^{(j)} w + 2\theta w^T h_j + \theta^2 h_{\tau\tau} - r^T r)}{\theta^2}. \tag{5.28}$$

5.4.2 Sparse case

Null-space representation In the large-scale case, A_j is partitioned into basic and superbasic variables [MS78]. Let

$$A_j = \begin{pmatrix} B & S \end{pmatrix},$$

where B is a square nonsingular basis. At a given iteration, the corresponding constraints are given by

$$A_j d = \begin{pmatrix} B & S \end{pmatrix} \begin{pmatrix} d_B \\ d_S \end{pmatrix} = 0.$$

A matrix Z that spans the null-space of A_j is

$$Z_j = \begin{pmatrix} -B^{-1}S \\ I \end{pmatrix}.$$

Sparse LU factors of B allow products $Z_j v$ and $Z_j^T w$ to be computed without storing Z_j explicitly. Note that the columns of Z_j are not orthonormal (or even orthogonal).

When a bound is deleted, one of the nonbasic variables becomes superbasic. The updated null-space matrix Z_{j+1} has the form

$$Z_{j+1} = \begin{pmatrix} -B^{-1}S & -B^{-1}s \\ I & 0 \\ & 1 \end{pmatrix}. \tag{5.29}$$

The relationship $A_{j+1}Z_{j+1} = 0$ is maintained.

Reduced Hessian Let $v \equiv -B^{-1}s$. In terms of v, the reduced Hessian in the new set of free variables is

$$Z_{j+1}^T H_{j+1}^{(j)} Z_{j+1} = \begin{pmatrix} Z_j^T H_j^{(j)} Z_j & Z_j^T H_j^{(j)} v + Z_j^T h_\tau \\ v^T H_j^{(j)} Z_j + h_\tau^T Z_j & v^T H_j^{(j)} v + 2v^T h_j + h_{\tau\tau} \end{pmatrix}. \tag{5.30}$$

If in the process of updating the Cholesky factors it emerges that $Z_{j+1}^T H_{j+1}^{(j)} Z_{j+1}$ is sufficiently positive definite then $H^{(j+1)} = H^{(j)}$.

If the reduced Hessian is not sufficiently positive definite, we modify the diagonal element $h_{\tau\tau}$ of the Hessian by adding $\delta_j^H > 0$ to it just as in the dense case (5.23), i.e.,

$$H^{(j+1)} = H^{(j)} + \delta_j^H e_\tau e_\tau^T.$$

It follows from (5.29) and (5.23) that

$$Z_{j+1}^T H_{j+1}^{(j+1)} Z_{j+1} = Z_{j+1}^T H_{j+1}^{(j)} Z_{j+1} + \delta_j^H e_{n_Z} e_{n_Z}^T. \tag{5.31}$$

The updated Cholesky factor also has the same form as in the dense case. The vector r of R_{j+1} is uniquely defined, given equations (5.30), (5.31), and (5.26), as the solution to

$$R_j^T r = Z_j^T H_j^{(j)} v + Z_j^T h_\tau. \tag{5.32}$$

If the last diagonal element ϱ of the Cholesky factor is strictly positive, then

$$r^T r + \varrho^2 = v^T H_j^{(j)} v + 2v^T h_j + h_{\tau\tau} + \delta_j^H.$$

Thus a sufficiently large δ_j^H for $\varrho > \epsilon$ is

$$\delta_j^H > \epsilon^2 - (v^T H_j^{(j)} v + 2v^T h_j + h_{\tau\tau} - r^T r). \tag{5.33}$$

5.4.3 Properties of the modification

For each of the desirable properties for the Hessian modification stated in Section 5.3.1, we show if and how it holds for the particular modification described above.

1. **Low rank change to H.** Our algorithm requires at most a rank-one change each time a bound is deleted from the active set.

2. **Positivity of reduced Hessian.** Since the updates to H only modify the matrix by the addition of a positive semi-definite matrix, it follows that if $Z_j^T H^{(r)} Z_j \succ 0$ then so is $Z_j^T H^{(s)} Z_j \succ 0$ for $s \geq r$. Hence, the form of the update guarantees that if the algorithm terminates after J iterations, $Z_j^T H^{(J-1)} Z_j \succ 0$ for $j = 0, \ldots, J-1$.

3. **Condition of reduced Hessian.** Our update allows for the reduced Hessian to be modified so $\varrho > \epsilon$ for some ϵ (see (5.25), (5.28), and (5.33)). By doing this we can ensure the minimum eigenvalue of the updated reduced Hessian is bounded away from zero.

4. **Constancy of reduced Hessian.** $Z_j^T H_j^{(j+1)} Z_j = Z_j^T H_j^{(j)} Z_j$ is true by design because the modification to $H_j^{(j)}$ is orthogonal to Z_j.

5. **Stationary point and multipliers.** Lemma 5.4.1 below states that p_j remains a stationary point after the modification to H. Regarding the multipliers, the π's do not change and the σ's may change after the Hessian modification, but only the one corresponding to the new free variable, and it can only decrease.

Conditions at stationary point and multipliers after Hessian modification

Let π_j be the multipliers for the constraints and σ_j be the multipliers for the bounds (both active and inactive) at the stationary point p_j, and let π_{j+1} and σ_{j+1} be the corresponding values after the Hessian has been modified.

Lemma 5.4.1 *Suppose the τth fixed variable is set free and the Hessian $H^{(j)}$ is modified to $H^{(j+1)}$ according to the formula in the previous section (either sparse or dense).*

Then p_j remains a constrained stationary point of the new problem. The updated multiplier values for the constraints stay the same ($\pi_j = \pi_{j+1}$), and the updated multipliers for the bounds are

$$\sigma_{j+1} = \sigma_j - \delta^H e_\tau e_\tau^T x \le \sigma_j.$$

Proof. In order to establish that p_j remains a stationary point for the new problem, we need to show that

$$Z_j^T g_j = Z_j^T \bar{g}_j = 0,$$

where

$$\bar{g}_j = \bar{P}_j^T (g + H^{(j+1)} p_j).$$

The difference between the two is

$$\varphi \equiv Z_j^T \bar{P}_j^T (H^{(j+1)} - H^{(j)}) p_j.$$

It remains to be shown that $\varphi = 0$.

Since for both the dense and the sparse case,

$$H^{(j+1)} = H^{(j)} + \delta_j^H e_\tau e_\tau^T,$$

it follows that

$$\varphi = Z_j^T \bar{P}_j^T (\delta^H e_\tau e_\tau^T) p_j = \delta^H Z_j^T \bar{P}_j^T e_\tau (p_j)_\tau.$$

Since $\bar{P}_j^T e_\tau = 0$, we have $\varphi = 0$.

To show the multipliers change only as specified, we first refer to the notation for partitioning of the variables into fixed and free defined in (2.9) and (2.10), and to the definition of the multipliers for the bounds in (2.11). Let FX, FR, and FRX be defined at the stationary point, *before* a bound is released.

The Hessian modification affects only H_{FX} and not H_{FR} or H_{FRX}. The multipliers corresponding to the general constraints are unaffected because the equation defining the multipliers,

$$A_{FR}^T \pi = g_{FR} + H_{FR} p_{FR} - H_{FRX} x_{FX},$$

is independent of H_{FX}.

Note that the only modified component affecting the σ's as defined in (2.11) is the term $H_{FX} x_{FX}$. The change to this term, projected into the full space, is $\delta_j^H e_\tau e_\tau^T x$. The multipliers σ_{j+1} corresponding to the bounds, projected into the full space, become

$$\begin{aligned} \sigma_{j+1} &= \sigma_j - \delta_j^H e_\tau e_\tau^T x \\ &\leq \sigma_j. \end{aligned}$$

Since $x \geq 0$, if $(\sigma_j)_\tau < 0$, then $(\sigma_{j+1})_\tau < 0$. If $(\sigma_j)_\tau$ was the smallest multiplier, it remains the smallest. ∎

5.4.4 Modified descent step

In Section 2.2.1, we described how to find a sufficient descent step d, from the first stationary point, that would satisfy conditions **DD1** and **DD2** given in Section 2.2.1. Even after multiple modifications to H, this algorithm for computing a descent step could also be applied at any "good" stationary point, a point $\tilde{p}$ satisfying conditions (5.17)–(5.19).

Here, we show that a sufficient descent step d satisfying **DD1** and **DD2** can also be computed by another algorithm—this one based on the next step(s) that would ordinarily be taken by an active-set method that did not stop at the "good" stationary point $\tilde{p}$. We show how d can be computed from the accumulation of steps taken from $\tilde{p}$ until the first "bad" stationary point. (A "bad" stationary point is one that does *not* satisfy all of conditions (5.17)–(5.19), or is unsatisfactory for another reason such as a failed attempt to modify H.)

Consider the very next step that would normally be taken by the active-set method after a "good" stationary point is reached. First, the variable corresponding to the minimum multiplier estimate would be released from its bound. Let the index of the released variable be $\tau = \arg\min_i(\tilde{\sigma}_{FX})_i$, and assume the corresponding multiplier estimate is strictly negative. Let $d_{FX'}$ be d restricted to the set of fixed variables excluding the newly freed τth variable. Let $\tilde{H}$ be the Hessian at $\tilde{p}$ (which may have been modified one or more times since $H^{(0)}$).

Direction without modifying $\tilde{H}$

After that variable is released, the next EQP (in the full variable space) is

$$\boxed{\begin{array}{ll} \underset{d}{\text{minimize}} & \frac{1}{2}d^T\tilde{H}d + \tilde{g}^T d \\ \text{subject to} & Ad = 0, \quad d_{FX'} = 0, \end{array}}$$

with the necessary conditions for optimality

$$\begin{aligned} \tilde{H}d - A^T\pi - \sigma_{FX} &= -\tilde{g} \\ Ad &= 0 \\ d_{FX'} &= 0. \end{aligned} \tag{5.34}$$

Suppose that without modifying the Hessian from $\tilde{H}$ there exists a solution $\bar{d}_N$ to (5.34). The following lemma states that a descent step can be derived from $\bar{d}_N$.

Lemma 5.4.2 *If a solution $\bar{d}_N$ exists to (5.34) then*

$$\tilde{g}^T(\bar{d}_N/\|\bar{d}_N\|_\infty) \le \left(\frac{\beta_\sigma \min_i \tilde{\sigma}_i}{n\|\bar{d}_N\|_\infty}\right)\tilde{g}^T u^*,$$

and the direction $\bar{d}_N/\|\bar{d}_N\|_\infty$ is a descent step satisfying conditions **DD1** *and* **DD2** *(see Section 2.2.1).*

Proof. It can be shown there exists $\beta_\sigma > 0$ such that if $\tilde{\sigma}_\tau < 0$ then $(\bar{d}_N)_\tau > -\beta_\sigma\tilde{\sigma}_\tau$. Since $\bar{d}_N$ satisfies (5.34),

$$A\bar{d}_N/\|\bar{d}_N\|_\infty = 0.$$

We show sufficient descent along $\bar{d}_N/\|\bar{d}_N\|_\infty$. Let $\tilde{g} = g_j$. Since at a stationary point

$\tilde{g} = A^T\tilde{\pi} + \tilde{\sigma}$ and $\tilde{\pi}^T A\bar{d}_N = 0$,

$$\begin{aligned}
\tilde{g}^T(\bar{d}_N/\|\bar{d}_N\|_\infty) &= (\tilde{\pi}^T A + \tilde{\sigma}^T)(\bar{d}_N/\|\bar{d}_N\|_\infty) \\
&= \tilde{\sigma}^T\bar{d}_N/\|\bar{d}_N\|_\infty \\
&= \tilde{\sigma}_\tau(\bar{d}_N)_\tau/\|\bar{d}_N\|_\infty \\
&= (-\tilde{\sigma}_\tau)(-(\bar{d}_N)_\tau)/\|\bar{d}_N\|_\infty \\
&\le (-\tilde{\sigma}_\tau)(\beta_\sigma\tilde{\sigma}_\tau)/\|\bar{d}_N\|_\infty \\
&= -\beta_\sigma(\min_i \tilde{\sigma}_i)^2/\|\bar{d}_N\|_\infty \\
&\le \left(\frac{\beta_\sigma \min_i \tilde{\sigma}_i}{n\|\bar{d}_N\|_\infty}\right)\tilde{g}^T u^*.
\end{aligned} \tag{5.35}$$

The last inequality follows from (2.18). ∎

Direction after modifying H

If the reduced Hessian is indefinite after $(d)_\tau$ is released from its bound, it may be necessary to modify $\tilde{H}$ to $\bar{H}$ to ensure a minimizer exists. Let $\bar{d}_N$ be the solution to the modified problem

$$\begin{aligned}
\bar{H}d - A^T\pi - \sigma_{FX} &= -\tilde{g} \\
Ad &= 0 \\
d_{FX'} &= 0.
\end{aligned} \tag{5.36}$$

Lemma 5.4.3 *If a solution $\bar{d}_N$ exists to (5.36) then*

$$\tilde{g}^T(\bar{d}_N/\|\bar{d}_N\|_\infty) \le \left(\frac{\beta_\sigma \min_i \tilde{\sigma}_i}{n\|\bar{d}_N\|_\infty}\right)\tilde{g}^T u^*,$$

and the direction $\bar{d}_N/\|\bar{d}_N\|_\infty$ is a descent step satisfying conditions **DD1** *and* **DD2**.

Proof. Let $\tilde{Z}$ be the null-space of the constraint matrix directly after the bound τ is dropped. $\bar{H}$ is chosen so $\tilde{Z}^T\bar{H}\tilde{Z} \succ 0$. By Lemma 5.4.1, the change made to the Hessian has no effect on the multipliers $\tilde{\pi}$ corresponding to the general constraints, and does not change the bounds multipliers $\tilde{\sigma}$ except for $\tilde{\sigma}_\tau$, and $\tilde{\sigma}_\tau$ does not increase but may decrease.

Denoting the new value by $\bar{\sigma}_\tau$, we note that $(\bar{d}_N)_\tau$ moves off its bound and satisfies

$$(\bar{d}_N)_\tau > -\beta_\sigma \bar{\sigma}_\tau > -\beta_\sigma \tilde{\sigma}_\tau.$$

Inequality (5.35) therefore holds and Lemma 5.4.2 is satisfied by the same argument as before. Hence $\bar{d}_N/\|\bar{d}_N\|_\infty$ is still a direction of sufficient descent even if $\tilde{H}$ needs to be modified. ∎

Additional steps

If one continues to minimize the QP (with the modified Hessian $\bar{H}$), the descent step $\bar{d}_N$ can be augmented by all steps taken until the next stationary point is reached. Let the sequence of directions be $\bar{d}_N, \bar{d}_{N+1}, \ldots, \bar{d}_{M-1}$, with feasible steplengths $\gamma_N, \gamma_{N+1}, \ldots, \gamma_{M-1}$, where $1 \geq \gamma_j > 0$. After the $M-N^{th}$ step $\bar{d}_{M-1}$, a new stationary point is reached.

Every direction $\bar{d}_j$ lies in the null-space $\tilde{Z}$ and can be written $\bar{d}_j = \tilde{Z}\bar{d}_j^Z$. Given

$$\bar{s} \equiv \sum_{j=N}^{M-1} \gamma_j \bar{d}_j \tag{5.37}$$

(the step to the next stationary point after $\tilde{p}$), there exists $\bar{s}^Z$ such that $\bar{s} = \tilde{Z}\bar{s}^Z$. It follows that

$$\bar{s}^T \bar{H} \bar{s} = (\bar{s}^Z)^T (\tilde{Z}^T \bar{H} \tilde{Z}) \bar{s}^Z > 0. \tag{5.38}$$

Lemma 5.4.4 $s \equiv \bar{s}/\|\bar{s}\|_\infty$ *is a descent step satisfying conditions* **DD1** *and* **DD2**.

Proof. We need to show there exists a constant $\kappa > 0$ such that $\tilde{g}^T s < \kappa \tilde{g}^T u^*$.

Since the QP objective $\psi(\bar{H})$ is strictly decreasing for $\gamma_N \bar{d}_N, \gamma_{N+1}\bar{d}_{N+1}, \ldots, \gamma_{M-1}\bar{d}_{M-1}$,

$$\tilde{g}^T \bar{s} + \frac{1}{2}\bar{s}^T \bar{H} \bar{s} < \gamma_N \tilde{g}^T \bar{d}_N + \frac{1}{2}\gamma_N^2 \bar{d}_N^T \bar{H} \bar{d}_N. \tag{5.39}$$

Moreover, since $\bar{d}_N$ is the optimal step on the first subspace,

$$\bar{d}_N^T \bar{H} \bar{d}_N = -\tilde{g}^T \bar{d}_N. \tag{5.40}$$

It follows from (5.38), (5.39), and (5.40) that

$$\begin{aligned}
\tilde{g}^T\bar{s} &< \gamma_N\tilde{g}^T\bar{d}_N + \frac{1}{2}\gamma_N^2\bar{d}_N^T\bar{H}\bar{d}_N - \frac{1}{2}\bar{s}^T\bar{H}\bar{s} \\
&< \gamma_N\tilde{g}^T\bar{d}_N + \frac{1}{2}\gamma_N^2\bar{d}_N^T\bar{H}\bar{d}_N \\
&= \gamma_N\tilde{g}^T\bar{d}_N - \frac{1}{2}\gamma_N^2\tilde{g}^T\bar{d}_N \\
&= \tilde{g}^T\bar{d}_N(\gamma_N - \frac{1}{2}\gamma_N^2). \qquad (5.41)
\end{aligned}$$

Since $0 < \gamma_N \leq 1$, it follows that $(\gamma_N - \frac{1}{2}\gamma_N^2) > 0$.

Finally, by Lemma 5.4.3 and (5.41),

$$\begin{aligned}
\tilde{g}^T s &< (\gamma_N - \frac{1}{2}\gamma_N^2)\tilde{g}^T\bar{d}_N/\|\bar{s}\|_\infty \\
&= (\gamma_N - \frac{1}{2}\gamma_N^2)\frac{\tilde{g}^T\bar{d}_N}{\|\bar{d}_N\|_\infty}\frac{\|\bar{d}_N\|_\infty}{\|\bar{s}\|_\infty} \\
&\leq (\gamma_N - \frac{1}{2}\gamma_N^2)\frac{\|\bar{d}_N\|_\infty}{\|\bar{s}\|_\infty}\left(\frac{\beta_\sigma \min_i \tilde{\sigma}_i}{n\|\bar{d}_N\|_\infty}\right)\tilde{g}^T u^*. \qquad (5.42)
\end{aligned}$$

Defining

$$\kappa \equiv (\gamma_N - \frac{1}{2}\gamma_N^2)\frac{\|\bar{d}_N\|_\infty}{\|\bar{s}\|_\infty}\left(\frac{\beta_\sigma \min_\tau \tilde{\sigma}_\tau}{n\|\bar{d}_N\|_\infty}\right)$$

gives the required result. ∎

5.5 Statement of the algorithm

In Figures 5.1–5.3, we provide pseudocode summarizing our new algorithm to compute the search direction from the indefinite QP subproblem. Algorithm xIDQP can be compared to IDQP presented in Section 2.2.1 of Chapter 2. As before, it is broken into three parts:

- Algorithm xIDQP: x-Compute-search-direction-from-indefinite-QP
- Subroutine x-Move-to-stationary-point
- Subroutine x-Compute-descent-direction

Algorithm xIDQP: x-Compute-search-direction-from-indefinite-QP

Obtain a feasible point p_0 satisfying (2.13)
Identify initial working set and compute P_0, A_0, Z_0, and H_0
$R \leftarrow$ modified Cholesky factor R (such that $\lambda_{\min}(R^TR = Z_0^T H_0^{(0)} Z_0) > \beta_{HZ}$)
$H^{(0)} \leftarrow H + M$ (M computed by (5.7) for dense, (5.10) for sparse case)
$j \leftarrow 0$
repeat
    $\boxed{\text{x-Move-to-stationary-point}}$
    Compute Lagrange multipliers σ_{FX} for the active bounds (cf. (2.11))
    if conditions (5.17)–(5.19) all hold **then**
        $\tilde{p} \leftarrow p_j$
        $\tilde{H} \leftarrow \tilde{H}^{(j)}$
        $\sigma_\tau \leftarrow \min_i \sigma_i$
        converged $\leftarrow \sigma_\tau \geq 0$
        if not *converged* **then**
            Delete bound with multiplier $\tilde{\sigma}_\tau$
            Update working set and compute P_{j+1}, A_{j+1}, and Z_{j+1}
            Compute $H^{(j+1)}$, $H_{j+1}^{(j+1)}$
            Update $R^TR = Z_{j+1}^T H_{j+1}^{(j+1)} Z_{j+1}$ and $H^{(j+1)}$
            $p_{j+1} \leftarrow p_j$
            $j \leftarrow j + 1$
        end
    else
        forced_stop $\leftarrow$ **true**
        $\boxed{\text{x-Compute-descent-direction}}$
    end
until *converged* **or** *forced_stop*

Figure 5.1: Algorithm xIDQP.

Subroutine x-Move-to-stationary-point

repeat

$g_j \leftarrow \bar{P}_j^T(g + H^{(j)}p_j)$

$\mathit{stationary\_point} \leftarrow Z_j^T g_j = 0$

if not $\mathit{stationary\_point}$ **then**

Solve for d satisfying $\begin{pmatrix} H_j^{(j)} & A_j^T \\ A_j & 0 \end{pmatrix} \begin{pmatrix} d \\ -\pi \end{pmatrix} = \begin{pmatrix} -g_j \\ 0 \end{pmatrix}$

$\bar{d} \leftarrow P_j^T \begin{pmatrix} d \\ 0 \end{pmatrix}$

$\gamma_\tau \leftarrow \min_i \left\{ -\dfrac{x_i + p_i}{\bar{d}_i} \mid \bar{d}_i < 0 \right\}$

$\mathit{hit\_constraint} \leftarrow \gamma_\tau < 1$

$\gamma \leftarrow$ **if** $\mathit{hit\_constraint}$ **then** γ_τ **else** 1

$p_{j+1} \leftarrow p_j + \gamma\bar{d}$

$H^{(j+1)} \leftarrow H^{(j)}$

Update working set and compute P_{j+1}, A_{j+1}, Z_{j+1}, and $H_{j+1}^{(j+1)}$

$j \leftarrow j + 1$

end

until $\mathit{stationary\_point}$

Figure 5.2: Subroutine x-Move-to-stationary-point.

Subroutine x-Compute-descent-direction

$\hat{u} \leftarrow p_j - \tilde{p}$

$u \leftarrow \hat{u}/\|\hat{u}\|$

if $-(g + \tilde{H}\tilde{p})^T u < \|\hat{u}\|(u^T\tilde{H}u)$, **then**

$\tilde{\gamma} \leftarrow \min\Big(-\dfrac{(g + \tilde{H}\tilde{p})^T u}{u^T\tilde{H}u}, \gamma_M\Big)$

else

$\tilde{\gamma} \leftarrow \min(\|\hat{u}\|, \gamma_M)$

end

if $\|\tilde{p} + \tilde{\gamma}u\| < \|\tilde{p}\|$, **then** $\tilde{\gamma} \leftarrow 0$

if $g^T u > 0$, **then** $\tilde{\gamma} \leftarrow 0$

Figure 5.3: Subroutine x-Compute-descent-direction.

5.6 Theoretical results

5.6.1 Finite termination of the QP algorithm

One consequence of multiple modifications to the Hessian is that we are repeatedly changing the QP being "solved". This raises the issue of whether xIDQP (see Figure 5.1) is guaranteed to terminate in a finite number of iterations. It is possible to show that some earlier constrained stationary points may no longer be constrained stationary points of the QP after the Hessian has been modified at a later stage of xIDQP, as described in Section 5.4. It may be necessary, therefore, to revisit a particular working set many times. Despite this, we show in the next lemma that xIDQP terminates after a finite number of iterations.

Lemma 5.6.1 *xIDQP terminates after a finite number of iterations.*

Proof. Assume xIDQP does not terminate. Then there exists a subsequence of iterations, say T, such that for $j \in T$ we have $H^{(j)} \neq H^{(j-1)}$. Let $\{\bar{t}_j\}$ be a subsequence of T such that the active set at the $\bar{t}_j^{th}$ iteration is constant. Such a subsequence must exist because there is a finite number of choices for the set of free variables. At iteration $\bar{t}_1$ the reduced Hessian is modified to make it positive definite. Although in the intervening iterations between $\bar{t}_1$ and $\bar{t}_2$ the Hessian may be modified, it is only modified by adding a positive-semidefinite matrix. It is, therefore, not necessary to modify the Hessian at iteration $\bar{t}_2$, which contradicts that this is an iteration where a modification was required. ∎

5.6.2 Global convergence properties

Theorem 5.6.1 *Algorithm IDSQP in combination with xIDQP has the property that*

$$\lim_{k\to\infty} \|x_k - x^*\| = 0,$$

where x^* is a KKT point for INP.

Proof.

If we set the direction of negative curvature equal to zero at every iteration and substitute quasi-Newton estimates of the Hessian of the Lagrangian for the exact second derivatives, Murray and Prieto's algorithm SQP2D [MP99] reduces to IDSQP and IDQP. It follows from

Theorem 3.2 and Corollary 3.2 of [MP99] that for algorithms IDSQP and IDQP:

$$\begin{aligned}
&\lim_{k\to\infty} \|\hat{p}_k\| = 0, \quad \text{and} \\
&\lim_{k\to\infty} \|x_k - x^*\| = 0, \quad \text{where } x^* \text{ is a KKT point for NP.}
\end{aligned} \tag{5.43}$$

It remains to show that when IDQP is replaced by xIDQP, these results still hold. Lemma 5.6.1 proves that xIDQP terminates after a finite number of iterations. We examine how the search direction from xIDQP, combining the step to a stationary point with a descent step off the stationary point, affects the proofs in [MP99]. The only lemmas from [MP99] significantly affected by the changes to the step to the stationary point and by the modifications to the Hessian within xIDQP are Lemma 3.4 (condition (5.11)) and Lemma 3.5 (condition (5.13)).

The proof of Lemma 3.4 relies on the general positivity assumption, i.e., that "the reduced Hessians of the modified QP are sufficiently positive definite on all the subsets of constraints encountered when determining a stationary point." Moving past the first stationary point may cause this assumption to be violated. It was shown in Section 5.3.3 that Lemma 3.4 is guaranteed to hold at the first stationary point. Because algorithm xIDQP checks condition (5.18) (equivalent to (5.11)) at every stationary point before proceeding, it ensures that Lemma 3.4 holds at the final stationary point $\tilde{p}$. For more details, please refer to the earlier discussion of Lemma 3.4 and Lemma 3.5 in Section 5.3.3.

Two points in the proof of Lemma 3.5 need to be justified in light of changes introduced by xIDQP. First, it is necessary to justify the bound (5.14), an upper bound on the directional positive curvature. Second, we need $(g + \tilde{H}p_j)^T \bar{d}_j < 0$ for all j. The latter condition holds as long as the Hessian has not been modified after the initial feasible point (i.e., $\tilde{H} = H^{(0)}$), in which case the third condition, (5.19), is equivalent to (5.14). Otherwise, the third condition is equivalent to the main result of Lemma 3.5, (5.13).

Finally, the descent step off the final stationary point, a step from the last "good" stationary point (satisfying (5.17)–(5.19)) to the next "bad" stationary point (violating one of the conditions) was shown to satisfy **DD1** and **DD2** in Lemmas 5.4.2–5.4.4. ∎

Chapter 6

Computational Results and Conclusions

In this chapter we present numerical results supporting the following hypotheses:

- The direct BFGS quasi-Newton approximation to the Hessian of the Lagrangian (discussed in Chapter 3) may perform poorly even on simple problems with quadratic functions, preventing a superlinear rate of convergence in the SQP algorithm.

- Superlinear convergence of the SQP method is attainable on the same problems using the disaggregated SR1 quasi-Newton Hessian approximation (presented in Chapter 4).

In Section 6.1, we describe the details of our MATLAB implementation of an SQP method based on subroutines IDSQP and xIDQP from Chapters 2 and 5. In the following sections, we introduce test problems and present corresponding results in graphical form. We finish by reviewing our hypotheses and summarizing the findings suggested by our results.

6.1 Implementation

The MATLAB code is based on a set of routines originally written by Murray and Prieto to implement their second-derivative algorithm SQP2D [MP99]. Their code constructed a search direction based upon the first constrained stationary point encountered in the QP subproblem (see algorithm IDQP, Chapter 2). Our code allows the option of continuing past the first stationary point and in some cases continuing all the way to a minimizer (see algorithm xIDQP, Chapter 5). We call the main program `sqpqn.m` (implements IDSQP), and we call the subprogram that computes the descent direction from the QP subproblem `iqp.m` (implements xIDQP). We compared the two quasi-Newton Hessian approximations, the positive-definite, direct BFGS Hessian approximation and the disaggregated SR1 Hessian approximation, using the MATLAB programs `sqpqn.m` and `iqp.m`. Recall that given a positive-definite Hessian approximation and well-defined and conditioned Jacobians, algorithm IDSQP reduces to the positive-definite SQP algorithm PDSQP and xIDQP reduces to PDQP.

The code was written solely for the purpose of testing the hypotheses and not as a robust efficient code for solving practical problems. Many details were left open in the description of the algorithms (e.g., "perform a linesearch") given and described in Chapters 2 and 5. When choices needed to be made for the implementation, we usually took the simplest path from a programming perspective.

Our MATLAB implementation uses dense linear algebra, except for the representation of the disaggregated quasi-Newton Hessian approximation. We make use of sparsity in storing, evaluating, and updating the disaggregated quasi-Newton approximation. The nonzero elements of the component Hessian matrices are concatenated and stored as one long vector. The full Hessian is computed once per major iteration, using the latest multiplier estimates. Sparse linear algebra techniques are not used within the QP subroutine `iqp.m`. When a system of equations needs to be solved, MATLAB's backslash operand is called (finding the least-squares solution, for the rectangular case). The nullspace matrix for the working set is computed with MATLAB's QR factorization, initially and every time a bound is added to the working set.

6.1.1 SQP main algorithm

Linesearch The linesearch routine uses cubic interpolation (with some safeguards) and is followed by a loop to limit the infeasibilities.

Penalty parameters A vector of penalty parameters is used, one for each constraint, rather than a constant ρ for all constraints. The algorithm for adjusting the vector ρ of penalty parameters is similar to that described by Eldersveld in [Eld91] and implemented in NPSOL [GMSW86]. The initial penalty parameters are set to $\rho = e$. To adjust ρ we set a threshold of $\frac{1}{2}\omega$ instead of ω, i.e., $\phi'(0) \leq -\frac{1}{2}\omega$ (see (2.22)). We maintain a minimum value of $\rho \geq \rho_{\min} = 10^{-6}$.

Initial multiplier estimates After identifying a working set at the initial point, we set the initial multiplier estimates λ to the least-squares solution of $\widehat{A}^T\lambda = \widehat{g}$, where $\widehat{A}$ is the constraint matrix in terms of the free variables and working set of constraints and $\widehat{g}$ is the gradient in terms of the free variables.

Quasi-Newton approximations The aggregate BFGS approximation was initialized to the identity matrix. The Powell modification rule was used to maintain positive definiteness.

The initial SR1 component approximations were initialized to zero, but the identity was used as the approximation for the Hessian of the Lagrangian for the first two iterations. Exceptions were handled according to the rules described in Chapter 4.

Infeasible subproblems There are good ways to handle infeasible QP subproblems (such as adding elastic variables; see [Bom99, GMS97]). However, we do not handle infeasible subproblems well. When a feasible point satisfying (2.13) cannot be found, `iqp.m` is skipped. The point p_0 found by the feasible point routine is used in the linesearch by setting $p = p_0$ and the multiplier estimates are left at their previous value.

Lagrange multiplier estimates The Lagrange multiplier estimates used for the linesearch are the multipliers at the last stationary point generated in the QP subproblem. In certain situations these estimates are first modified: when one is negative, it is changed to zero, and when the norm of the search direction in the Lagrange multipliers or the QP multiplier estimate itself becomes too large, the whole vector is scaled down. The multiplier estimates used for the quasi-Newton update at the end of each major iteration are then taken from the result of the linesearch.

Convergence test To compute the reduced gradient for the convergence test, the working set (active constraints and active bounds) is inherited from the search direction subroutine `iqp.m`. Then the QR factorization is applied to compute Z.

6.1.2 Search direction subroutine

Feasible point The feasible point routine we use starts with a point satisfying the upper and lower bounds. It then proceeds to use an active-set method to minimize the sum of the infeasibilities, subject to the constraints and bounds currently satisfied. The objective and constraints are modified each time a new constraint becomes satisfied, until all constraints and bounds are satisfied.

It is preferable to arrive at the "correct" active set for the initial feasible point (i.e., the active set at the solution of the original nonlinear problem) so that eventually the initial reduced Hessian in `iqp.m` will be positive definite and will not need to be modified. To initialize the feasible point routine, we adjusted the free variables (those some distance from their bounds) by computing the shortest step to satisfy the active constraints. Close to the solution, this will be a feasible step. Away from the solution we may need to modify this step to satisfy the bounds.

Problem format At every major iteration the QP is converted from the inequality-constrained form (IQP) to the equality-constrained form (EQP) by adding slack variables to the inequality constraints.

Modified Cholesky The modified Cholesky algorithm is used to compute an initial modified reduced Hessian; see [GMW81, pp. 111]. The performance of the disaggregated SR1 method is sensitive to the choice of the parameter ϵ_R designating the minimum value along the diagonal of the Cholesky factor. We set ϵ_R to be an affine function of the largest element of the reduced Hessian (in absolute value). In fact, we used two different affine functions to allow ϵ_R to be smaller in the neighborhood of the solution (determined by whether the constraint infeasibilities were small).

Bounded Hessian There is a four-part strategy for keeping the Hessian matrices bounded. First, measures are taken to ensure the quasi-Newton updates are bounded so the Hessian components remain bounded. Second, the multipliers used to form the full Hessian (once per major iteration) are scaled back if necessary to ensure boundedness. Third, the modified Cholesky factorization used to compute the initial modified Hessian in `iqp.m` produces only bounded modifications to the Hessian. Fourth, the QP is terminated early if further modifications to the Hessian (see Section 5.4) are too large.

Degeneracy and working sets To choose a working set, we simply identify variables close to their bounds. The algorithm we have implemented in MATLAB assumes no degeneracy in the resulting working-set matrix will occur. When a bound is deleted from the working set, we solve a system of equations to determine the modified nullspace matrix. If the working-set matrix is degenerate or near-degenerate, this system of equations is ill-conditioned and there may be difficulties updating Z. In such a case we may resort to terminating at the current stationary point and computing a descent step (in the manner of Figure 2.6).

6.2 Test problems

Hanging chain and springs problems

The hanging chain problem appeared as an example in [Lue84]. A chain, consisting of n links of stiff steel, is suspended from two hooks that are w feet apart on a horizontal line. Each link is one foot in length. The problem is to determine the equilibrium shape of the chain by minimizing the potential energy of the chain.

Let link j span a horizontal distance of x_j and a vertical distance of y_j. Then $x_j^2+y_j^2=1$. The potential energy of a link is its weight times its vertical height (from some reference) times the gravitational constant g (approximately 9.8). The potential energy of the chain is the sum of the potential energies of each link. Take the top of the chain as reference and assume that the mass of each link is one unit, concentrated at its center. The potential energy is then

$$F(y) = g\ (\frac{1}{2}y_1 + (y_1+\frac{1}{2}y_2) + \ldots + (\sum_{j-1}^{n-1} y_j + \frac{1}{2}y_n)) = g\sum_{j=1}^{n}(n+\frac{1}{2}-j)\ y_j.$$

The chain is subject to two constraints: the total vertical displacement is zero, and the total horizontal displacement is w. The vertical displacement being zero means $\sum_j y_j = 0$, which implies

$$F(y) = g\sum_{j=1}^{n}(n+\frac{1}{2}-j)\ y_j = -g\sum_{j=1}^{n} j\ y_j.$$

Thus, the equilibrium shape is the solution of

$$
\begin{array}{ll}
\underset{x,y}{\text{minimize}} & F(y) = -g \sum_{j=1}^{n} j\, y_j \\
\text{subject to} & \sum_{j=1}^{n} y_j = 0 \\
& \sum_{j=1}^{n} x_j = w \\
& x_j^2 + y_j^2 = 1, \quad j = 1, \ldots, n.
\end{array}
$$

(Subtracting a multiple of an equality constraint from the objective affects the optimal multipliers, but not x^* or y^*.)

The hanging chain problem can be solved in closed form [Lue84, pp. 304]. However, depending on the formulation and the size of n and w, the problem can be quite challenging for SQP software. Moreover, there are variations of this problem that cannot be solved in closed form.

Before introducing one such variation, we introduce a slightly different formulation of the chain problem. Let (x_j, y_j) be the coordinates at the right node-end, for $j = 1, \ldots, n-1$ (so x_j is the horizontal distance from the left end of the chain, not just the horizontal displacement of one link). We define the constants $x_0 = 0$ and $y_0 = 0$, so the left end of the chain is fixed at the point $(0, 0)$. The linear inequality constraints from the previous formulation are satisfied by setting $y_n = 0$ and $x_n = w$ to be fixed values. The resulting problem formulation is

$$
\begin{array}{ll}
\underset{x,y}{\text{minimize}} & F(y) = g \sum_{j=1}^{n-1} y_j \\
\text{subject to} & (x_j - x_{j-1})^2 + (y_j - y_{j-1})^2 = 1, \quad 1 \le j \le n-1 \\
& (w - x_{n-1})^2 + (y_{n-1})^2 = 1 \\
& x \ge 0, \qquad y \le 0.
\end{array}
$$

The hanging springs problem is to find the shape of a hanging chain, where each link is a spring. In addition, each node may have a weight hanging from it. This problem is based on an AMPL model obtained from Robert Vanderbei and originally appeared in [LVBL98].

As before, let the variables (x_j, y_j) be the coordinates of the nodes, for $j = 1, \ldots, n-1$.

Let t_j, $j = 1, \ldots, n$, be the nonnegative extension of each spring. By default the mass of each node is 1, and the resting length of each spring is 1. Let the stiffness of the springs be described by the constant $k = 100$ and the gravitational constant be denoted g. Fix $x_0 = 0$, $y_0 = 0$, $y_n = 0$, and $x_n = w$. The resulting problem formulation has $3n - 2$ variables and n inequality constraints:

$$\boxed{\begin{array}{lll} \underset{x,y}{\text{minimize}} & g \displaystyle\sum_{j=1}^{n-1} y_j + \frac{k}{2} \sum_{j=1}^{n} t_j^2 & \\ \text{subject to} & (t_j+1)^2 - (x_j - x_{j-1})^2 - (y_j - y_{j-1})^2 \geq 0, & j = 1, \ldots, n \\ & x \geq 0, \qquad y \leq 0, \qquad t \geq 0. & \end{array}}$$

For a starting point, let

$$\begin{aligned} x_j &= \frac{jw}{n} && \text{for } j = 1, \ldots, n-1 \\ y_j &= d\left(-\frac{n}{2} + |j - \frac{n}{2}|\right) && \text{for } j = 1, \ldots, n-1 \\ t_j &= 0 && \text{for } j = 1, \ldots, n, \end{aligned}$$

where

$$d = \sqrt{1 - (w/n)^2}.$$

We ran several problem instances for different values of n and w. For each problem instance, we present a pair of graphs. The graph on the left shows how the norm of the infeasibilities converges to zero as the iterations increase. The one on the right shows $\|\nabla\mathcal{L}\|$, a measure of the convergence to optimality, versus number of iterations. To show the fullest range of values along the vertical axis, we use log plots, and on each plot overlay results from two different SQP algorithms:

1. The traditional algorithm, PDSQP, using the direct BFGS Hessian approximation and solving the positive-definite QP subproblem to optimality.

2. Algorithm IDSQP using the disaggregated SR1 Hessian approximation and computing the search direction with xIDQP.

The problem instances are described in Table 6.1. The number of degrees of freedom at the solution is in the column labelled *nd*. Computational results are displayed in Figures

Problem #	n	w	variables	constraints	nd
1	12	11	34	12	22
2	24	12	70	24	46
3	40	20	118	40	78

Table 6.1: Hanging springs problem instances.

6.1–6.3. We see that the disaggregated SR1 Hessian approximation performs significantly better than the direct BFGS approximation. Many fewer iterations are needed, and only the disaggregated method has a superlinear asymptotic rate of convergence.

We discussed in Chapter 3 that the aggregate BFGS update might work well if the infeasibilities converge quickly and the final search directions are in the null-space of the constraints. For the infeasibilities to converge at a quadratic rate, it is critical that the unit step be taken in the linesearch, but on the hanging springs problem, unit steps were not taken until after many iterations. The search direction in the nullspace of the constraints was so poor that it inhibited the search direction in the constraints, preventing rapid convergence of the infeasibilities. When the algorithm finally begins to take unit steps, the plot of the infeasibilities turns sharply downward, and the reduced gradient begins to converge faster as well.

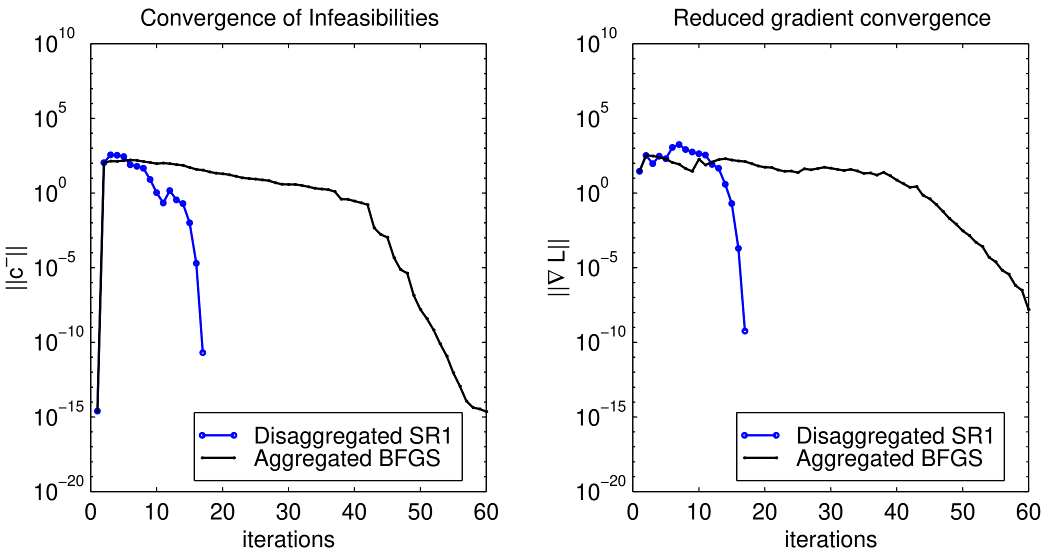

Figure 6.1: Results for hanging springs problem 1.

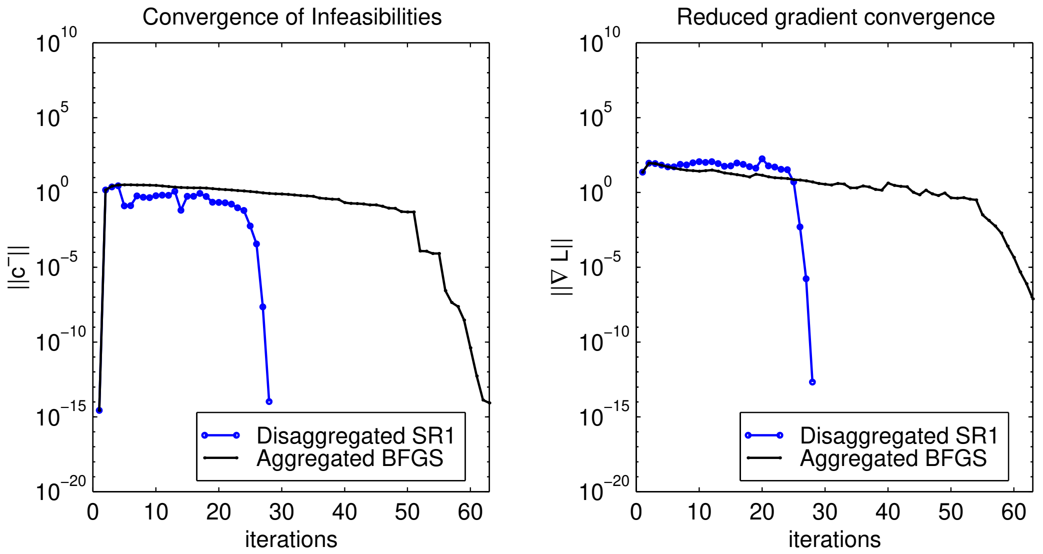

Figure 6.2: Results for hanging springs problem 2.

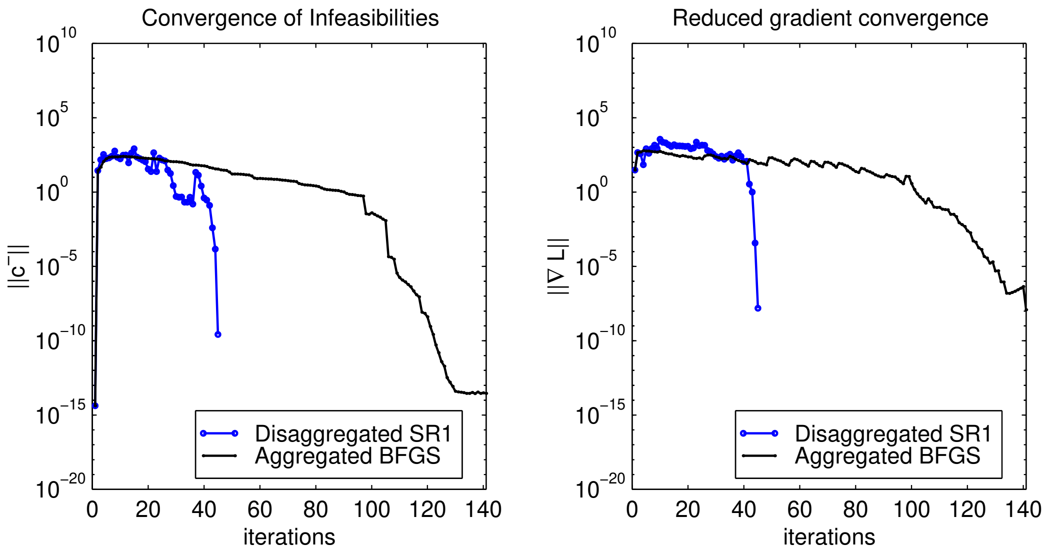

Figure 6.3: Results for hanging springs problem 3.

On this class of test problems the constraints are no more nonlinear than quadratic, so we would expect the disaggregated SR1 Hessian approximation to perform particularly well. At the same time, one would hope the direct BFGS approximation would also perform well on such simple examples.

Consider the following non-quadratic formulation of the hanging springs problem:

$$\begin{array}{ll} \underset{x,y}{\text{minimize}} & g\sum_{j=1}^{n-1} y_j + \frac{k}{2}\sum_{j=1}^{n} t_j^2 \\ \text{subject to} & (t_j+1) \geq \sqrt{(x_j - x_{j-1})^2 + (y_j - y_{j-1})^2}, \quad j = 1,\ldots,n. \\ & x \geq 0, \qquad y \leq 0. \end{array}$$

For this formulation, the Jacobian is highly nonlinear, as is the Hessian of the Lagrangian. Computational results displayed in Figures 6.4–6.6 show that both algorithms take slightly longer but their relative performance is about the same as before.

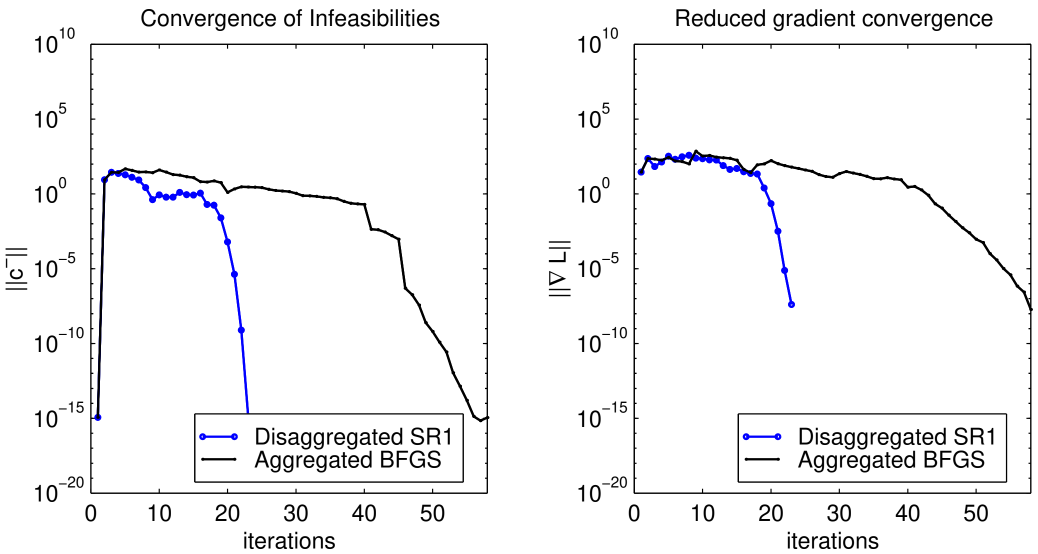

Figure 6.4: Results for hanging springs non quadratic formulation, problem 1.

Rocket fastest trip problem

The rocket fastest trip problem is based on an AMPL model obtained from Robert Vanderbei. The objective of this problem is to minimize the total time T that it takes for a

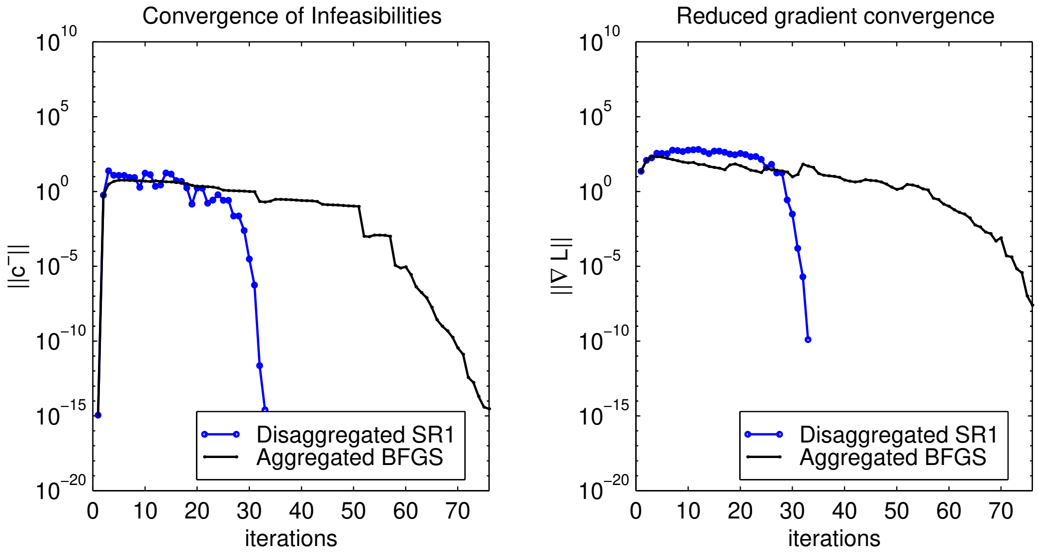

Figure 6.5: Results for hanging springs non quadratic formulation, problem 2.

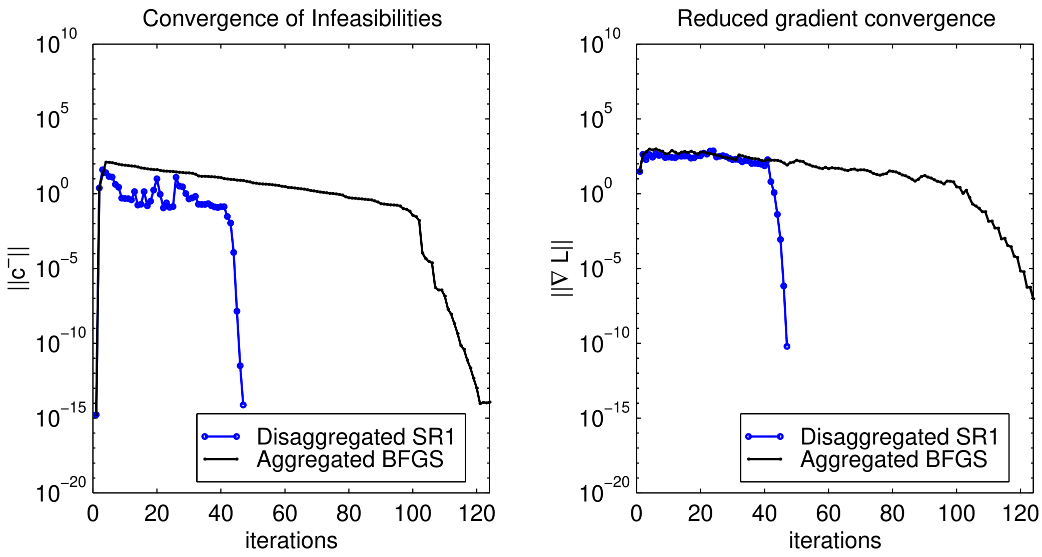

Figure 6.6: Results for hanging springs non quadratic formulation, problem 3.

rocket to get from an initial point to the final destination. Time is broken up into n equal intervals summing to T. The variables are

- x_j, the position of the rocket at time jT/n for $j = 1, \ldots, n-1$
- v_j, the velocity at time $(j+0.5)T/n$ for $j = 1, \ldots, n-2$
- a_j, the acceleration at time jT/n for $j = 1, \ldots, n-1$
- T, the total flight time.

Note that x_0, x_n, v_0, and v_{n-1} are *not* variables. The initial and final position (x_0, x_n) and the initial and final velocity (v_0, v_{n-1}) are boundary conditions (constants). There are upper and lower bounds on velocity and acceleration: $v_{\min}, v_{\max}$ and $a_{\min}, a_{\max}$. The rocket position is not bounded above or below. Nonlinear equalities define velocity and acceleration at each time point.

$$
\begin{array}{lll}
\underset{x,v,a,T}{\text{minimize}} & T & \\
\text{subject to} & n(x_j - x_{j-1}) = v_{j-1}T & j = 1, \ldots, n \\
& n(v_j - v_{j-1}) = a_j T & j = 1, \ldots, n-1 \\
& v_{\min} \le v_j \le v_{\max} & j = 1, \ldots, n-2 \\
& a_{\min} \le a_j \le a_{\max} & j = 1, \ldots, n-1 \\
& T \ge 0 &
\end{array}
$$

The problem instances we consider vary over n. For all of them we set

$$
\begin{array}{rclcrcl}
v_{\max} & = & 5 & \qquad & x_0 & = & 0 \\
v_{\min} & = & -5 & & x_n & = & 100 \\
a_{\max} & = & 1 & & v_0 & = & 0 \\
a_{\min} & = & -1 & & v_n & = & 0.
\end{array}
$$

All variables are initialized at 0.1 except for the final time T, which is initialized at 100.

The problem instances are described in Table 6.2. The results are displayed in Figures 6.7–6.8. The test results here are very different from the hanging springs problem because the solution is at a vertex. The reduced gradient is zero at any vertex. Once

Problem #	n	variables	constraints
1	30	87	59
2	40	117	79

Table 6.2: Rocket fastest trip problem instances.

the correct working set is identified, the constraint infeasibilities should converge to zero quadratically, since both methods reduce to Newton's method applied to a set of equations. However, it can be seen that the algorithm using the disaggregated SR1 Hessian approximation still performs significantly better than the one using the direct BFGS approximation, as the correct vertex is identified sooner.

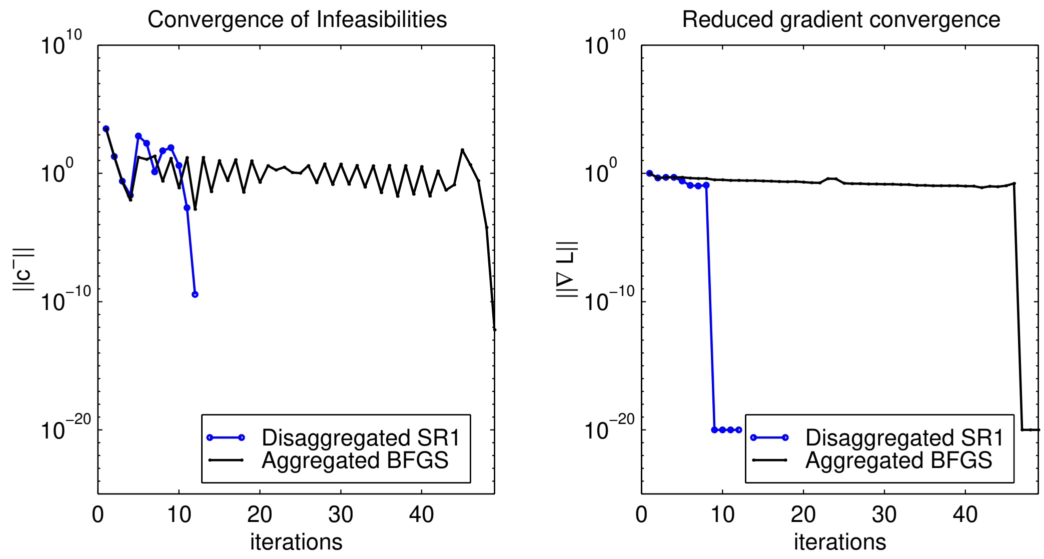

Figure 6.7: Results for rocket problem 1.

6.3 Conclusions

From the experimental results, we see that the standard direct BFGS quasi-Newton approximation to the Hessian of the Lagrangian does not work especially well even on these simple quadratic examples. As described in Chapter 3, the direct approximation uses a single update matrix, assumes the multiplier estimates are constant, and maintains positive-definiteness using the BFGS update. Treating the multiplier estimates as constant when

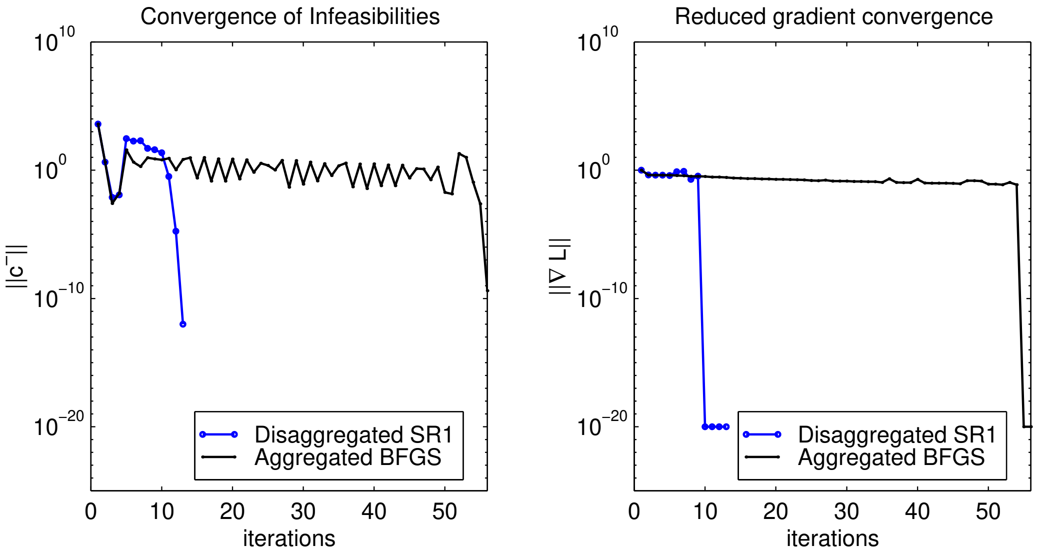

Figure 6.8: Results for rocket problem 2.

they actually vary as functions of x impacts the accuracy of the approximation. If eventually the null-space component of the search direction dominates the range-space component, the error of the approximation is mitigated. However, many iterations may be required for this to happen, if at all.

We developed a new technique for forming the quasi-Newton approximation, based on disaggregating $\nabla_x^2 \mathcal{L}(x, \lambda)$ into its component functions and using the SR1 update for each component. The SR1 update in combination with the disaggregated approach has many beneficial properties. It is a rank-one rather than a rank-two update. The multiplier estimates are no longer assumed constant. On quadratic functions the SR1 update can be proven to converge quickly. In the large-scale case this approximation can be updated and stored efficiently. This method also has the advantage that exact second derivatives can be incorporated into the update.

The challenge is how to work with an approximation of $\nabla_x^2 \mathcal{L}(x, \lambda)$ that is more accurate but perhaps not positive definite. The algorithm of Murray and Prieto handles such Hessian approximations, but mandates terminating at the first stationary point on all QP subproblems. There are certain advantages to reaching a solution of a QP subproblem. At the minimizer, one has a better guess at the active set for the original nonlinear program, and the multiplier estimates are positive on the active constraints. We cannot be worse off and we may be much better off by allowing the option of continuing, in some cases all the way to the solution of the indefinite QP subproblem.

The method we developed to address this challenge enables the algorithm of Murray and Prieto to proceed past the first stationary point of the QP subproblem. We have shown it is not trivial to decide how long one may continue past the first stationary point, since even a minimizer of a nonconvex QP may not lead to a sufficient descent direction for the merit function. The rules we developed always lead to a sufficient descent direction, and if the QP subproblem is strictly convex they allow the QP minimizer to be reached with no modification. Our algorithm does not modify the QP subproblem in the neighborhood of the solution even when the Hessian is indefinite. Consequently, when the approximation to $\nabla_x^2 \mathcal{L}(x, \lambda)$ is good, we can hope an SQP algorithm based on IDSQP and xIDQP will achieve a fast asymptotic rate of convergence.

Bibliography

[AS58] K. Arrow and R. M. Solow. Gradient methods for constrained maxima, with weakened assumptions. In L. Hurwicz K. J. Arrow and H. Uzawa, editors, *Studies in Linear and Non-Linear Programming*, pages 229–235. Stanford University Press, Stanford, CA, 1958.

[Ber82] D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods.* Academic Press, New York and London, 1982. ISBN 0-12-093480-9.

[BH93] J. T. Betts and W. P. Huffman. Path-constrained trajectory optimization using sparse sequential quadratic-programming. *J. of Guidance, Control, and Dynamics*, 16:59–68, 1993.

[BL85] A. Buckley and A. LeNir. BBVSCG: A variable-storage algorithm for function minimization. *ACM Trans. Math. Software*, 11:103–119, 1985.

[BNS94] R. H. Byrd, J. Nocedal, and R. B. Schnabel. Representations of quasi-Newton matrices and their use in limited-memory methods. *Math. Prog.*, 63:129–156, 1994.

[BNS95] L. T. Biegler, J. Nocedal, and C. Schmid. A reduced Hessian method for large-scale constrained optimization. *SIAM J. Optim.*, 5:314–347, 1995.

[Bom99] E. G. Boman. *Infeasibility and Negative Curvature in Optimization.* PhD thesis, Scientific Computing and Computational Mathematics Program, Stanford University, Stanford, CA, 1999.

[Bro70] C. G. Broyden. The convergence of a class of double-rank minimization algorithms, parts I and II. *J. Inst. Math. Applics.*, 6:76–90, 222–236, 1970.

[BSS93] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear programming: theory and algorithms.* John Wiley and Sons, New York, Chichester, Brisbane, Toronto and Singapore, second edition, 1993. ISBN 0-471-55793-5.

[BTW82] P. T. Boggs, J. W. Tolle, and P. Wang. On the local convergence of quasi-Newton methods for constrained optimization. *SIAM J. Control Optim.*, 20:161–171, 1982.

[CGT88] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. Testing a class of methods for solving minimization problems with simple bounds on the variables. *Math. Comput.*, 50:399–430, 1988.

[CGT91] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. Convergence of quasi-Newton matrices generated by the symmetric rank one update. *Math. Prog.*, 50:177–195, 1991.

[DS83] J. E. Dennis, Jr. and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1983.

[Eld91] S. K. Eldersveld. *Large-Scale Sequential Quadratic Programming Algorithms.* PhD thesis, Department of Operations Research, Stanford University, Stanford, CA, 1991.

[FGM95] A. Forsgren, P. E. Gill, and W. Murray. Computing modified Newton directions using a partial Cholesky factorization. *SIAM J. on Scientific Computing*, 16:139–150, 1995.

[Fle74] R. Fletcher. Methods related to Lagrangian functions. In P. E. Gill and W. Murray, editors, *Numerical Methods for Constrained Optimization*, pages 219–239. Academic Press, London and New York, 1974.

[Fle81] R. Fletcher. *Practical Methods of Optimization.* Volume 2: Constrained Optimization. John Wiley and Sons, Chichester and New York, 1981.

[Fle87] R. Fletcher. *Practical Methods of Optimization.* John Wiley and Sons, Chichester, New York, Brisbane, Toronto and Singapore, second edition, 1987.

[FM68] A. V. Fiacco and G. P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques.* John Wiley and Sons, Inc., New York, London, Sydney and Toronto, 1968.

[FM93] A. Forsgren and W. Murray. Newton methods for large-scale linear equality-constrained minimization. *SIAM J. Matrix Anal. Appl.*, 14:560–587, 1993.

[GL89] J. Ch. Gilbert and C. Lemaréchal. Some numerical experiments with variable-storage quasi-Newton algorithms. *Math. Prog.*, 45:407–435, 1989.

[GM74] P. E. Gill and W. Murray. Newton-type methods for unconstrained and linearly constrained optimization. *Math. Prog.*, 7:311–350, 1974.

[GM79] P. E. Gill and W. Murray. The computation of Lagrange multiplier estimates for constrained minimization. *Math. Prog.*, 17:32–60, 1979.

[GMS97] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. Report SOL 97-3, Department of Engineering-Economic Systems and Operations Research, Stanford University, Stanford, CA, 1997.

[GMSW86] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. User's guide for NPSOL (Version 4.0): a Fortran package for nonlinear programming. Report SOL 86-2, Department of Operations Research, Stanford University, Stanford, CA, 1986.

[GMSW89] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. Constrained nonlinear programming. In G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, editors, *Handbooks in Operations Research and Management Science, volume 1. Optimization*, chapter 3, pages 171–210. North Holland, Amsterdam, New York, Oxford and Tokyo, 1989.

[GMSW92] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. Some theoretical properties of an augmented Lagrangian merit function. In P. M. Pardalos, editor, *Advances in Optimization and Parallel Computing*, pages 101–128. North Holland, North Holland, 1992.

[GMW81] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press, London and New York, 1981. ISBN 0-12-283952-8.

[Goo85] J. Goodman. Newton's method for constrained optimization. *Math. Prog.*, 33:162–171, 1985.

[GT82a] A. Griewank and Ph. L. Toint. On the unconstrained optimization of partially separable functions. In M. J. D. Powell, editor, *Nonlinear Optimization 1981*, pages 301–312. Academic Press, London, 1982.

[GT82b] A. Griewank and Ph. L. Toint. Partitioned variable metric updates for large structured optimization problems. *Numer. Math.*, 39:119–137, 1982.

[GV89] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, Maryland, second edition, 1989.

[Hes69] M. R. Hestenes. Multiplier and gradient methods. *J. Optim. Theory and Applics.*, 4:303–320, 1969.

[KBS93] H. F. Khalfan, R. H. Byrd, and R. B. Schnabel. A theoretical and experimental study of the symmetric rank-one update. *SIAM J. Optim.*, 3:1–24, 1993.

[Leo95] M. W. Leonard. *Reduced Hessian Quasi-Newton Methods for Optimization*. PhD thesis, Department of Mathematics, University of California, San Diego, 1995.

[Lue84] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley Publishing Company, Reading, Menlo Park, London, Amsterdam, Don Mills and Sydney, 1984.

[LVBL98] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1998.

[MP95] W. Murray and F. J. Prieto. A sequential quadratic programming algorithm using an incomplete solution of the subproblem. *SIAM J. Optim.*, 5:590–640, 1995.

[MP99] W. Murray and F. J. Prieto. A second-derivative method for nonlinearly constrained optimization. Report SOL 95-3, Department of Operations Research, Stanford University, Stanford, CA, Revised 1999.

[MS78] B. A. Murtagh and M. A. Saunders. Large-scale linearly constrained optimization. *Math. Prog.*, 14:41–72, 1978.

[MS82] B. A. Murtagh and M. A. Saunders. A projected Lagrangian algorithm and its implementation for sparse nonlinear constraints. *Math. Prog. Study*, 16:84–117, 1982.

[MS95] B. A. Murtagh and M. A. Saunders. MINOS 5.4 User's Guide. Report SOL 83-20R, Department of Operations Research, Stanford University, Stanford, CA, Revised 1995.

[Mur69] W. Murray. An algorithm for constrained minimization. In R. Fletcher, editor, *Optimization*, pages 247–258. Academic Press, London and New York, 1969.

[Mur97] W. Murray. Sequential quadratic programming methods for large-scale problems. *Computational Optimization and Applications*, 7:127–142, 1997.

[Noc80] J. Nocedal. Updating quasi-Newton matrices with limited storage. *Math. Comput.*, 35:773–782, 1980.

[Pow69] M. J. D. Powell. A method for nonlinear constraints in minimization problems. In R. Fletcher, editor, *Optimization*, pages 283–298. Academic Press, London and New York, 1969.

[Pow77] M. J. D. Powell. A fast algorithm for nonlinearly constrained optimization calculations. Technical Report 77/NA 2, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, England, 1977.

[Pow78a] M. J. D. Powell. Algorithms for nonlinear constraints that use Lagrangian functions. *Math. Prog.*, 14:224–248, 1978.

[Pow78b] M. J. D. Powell. The convergence of variable metric methods for nonlinearly constrained optimization calculations. In O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, editors, *Nonlinear Programming 3*, pages 27–63. Academic Press, London and New York, 1978.

[Pow83] M. J. D. Powell. Variable metric methods for constrained optimization. In A. Bachem, M. Grötschel, and B. Korte, editors, *Mathematical Programming: The State of the Art*, pages 288–311. Springer Verlag, London, Heidelberg, New York and Tokyo, 1983.

[Rob72] S. M. Robinson. A quadratically-convergent algorithm for general nonlinear programming problems. *Math. Prog.*, 3:145–156, 1972.

[Van82] G. Van der Hoek. Asymptotic properties of reduction methods applying linearly equality constrained reduced problems. *Math. Prog. Study*, 16:162–189, 1982.

[Wil63] R. B. Wilson. *A simplicial method for convex programming.* PhD thesis, Harvard University, 1963.

[Wol62] P. Wolfe. The reduced-gradient method. Unpublished manuscript, the RAND Corporation, 1962.

[ZBLN97] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization. *ACM Trans. Math. Software*, 23(4):550–560, December 1997.