Title
Active-set methods for quadratic programming

Author
Wong, Elizabeth Lai Sum

Publication Date
2011

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA, SAN DIEGO

Active-Set Methods for Quadratic Programming

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

in

Mathematics

by

Elizabeth Wong

Committee in charge:

Professor Philip E. Gill, Chair
Professor Henry D. I. Abarbanel
Professor Randolph E. Bank
Professor Michael J. Holst
Professor Alison L. Marsden

2011

The dissertation of Elizabeth Wong is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California, San Diego

2011

TABLE OF CONTENTS

Signature Page iii

Table of Contents iv

List of Figures vi

List of Tables vii

List of Algorithms viii

Acknowledgements ix

Vita and Publications x

Abstract of the Dissertation xi

1 Introduction 1
  1.1 Overview 1
  1.2 Contributions of this Thesis 3
  1.3 Notation, Definitions, and Useful Results 4

2 Quadratic Programming 7
  2.1 Background 7
  2.2 Active-Set Methods for Mixed-Constraint Problems 9
    2.2.1 Binding-direction method 12
    2.2.2 Nonbinding-direction method 20
    2.2.3 Relation between the binding and nonbinding methods 28

3 Problems in Standard Form 30
  3.1 Introduction 30
  3.2 Nonbinding-Direction Method for Standard-Form QP 31
  3.3 Linear Programs in Standard Form 41

4 Dual Quadratic Programming 42
  4.1 Background 43
    4.1.1 Regularized dual problem 44
  4.2 A Dual Nonbinding-Direction Method 45
    4.2.1 Dual linear programming 56
    4.2.2 Degeneracy of the dual QP 56

5 Finding an Initial Point 57
  5.1 Getting Feasible 57
  5.2 Second-Order-Consistent Basis 58
    5.2.1 Variable-reduction method 59
    5.2.2 Schur-complement and block-LU method 59
  5.3 Stationarity 61

6 Single-Phase Methods . . . 63
6.1 Penalty-Function Methods . . . 63
6.2 QP Regularization . . . 65
6.3 Inner Iterations . . . 69
6.3.1 Constraint-shifted approach . . . 69
6.3.2 Variable-shifted approach . . . 72
6.4 Outer Iterations . . . 79

7 Solving the KKT Systems . . . 81
7.1 Variable-Reduction Method . . . 81
7.1.1 Equations for the standard-form algorithm . . . 82
7.2 Schur-Complement and Block-LU Method . . . 84
7.2.1 Augmenting the KKT matrix . . . 84
7.2.2 Factoring the matrices . . . 85
7.2.3 Updating the factors . . . 87

8 Numerical Results . . . 88
8.1 Implementation . . . 88
8.2 Performance Profiles . . . 90
8.3 Results for the Nonbinding-Direction Method . . . 91
8.3.1 Results on icQP with different linear solvers . . . 92
8.3.2 Results on icQP and SQOPT . . . 93

A Test Problem Data . . . 118

Bibliography . . . 121

LIST OF FIGURES

Figure 2.1: Sequence of subspace minimizer for the binding-direction method 18
Figure 2.2: Sequence of subspace minimizer for the nonbinding-direction method 26

Figure 3.1: Subspace minimizer sequence for the nonbinding-direction method 39

Figure 8.1: Performance profile of solve times for icQP on the CUTEr QP test set. 92
Figure 8.2: Performance profile of solve times for icQP on the Maros and Mészáros QP test set. 93
Figure 8.3: Performance profile of solve times for SQOPT and icQP on 55 convex CUTEr QPs with a small number of degrees of freedom. 96
Figure 8.4: Performance profile of solve times for SQOPT and icQP on 115 Maros and Mészáros QPs with a small number of degrees of freedom. 96
Figure 8.5: Performance profile of solve times for SQOPT and icQP on 35 convex CUTEr QPs with a large number of degrees of freedom. 97
Figure 8.6: Performance profile of solve times for SQOPT and icQP on 21 Maros and Mészáros QPs with a large number of degrees of freedom. 97
Figure 8.7: Performance profile of solve times for SQOPT and icQP–HSL_MA57 on 56 convex CUTEr and Maros and Mészáros QPs with a large number of degrees of freedom. 98

LIST OF TABLES

Table 8.1: Number of temporary constraints for icQP with HSL_MA57 90
Table 8.2: Results on a subset of problems from the CUTEr set for icQP–HSL_MA57 and icQP–HSL_MA57v 94
Table 8.3: Results for CUTEr QPs 99
Table 8.4: Results for Maros and Mészáros QPs 104
Table 8.5: Number of superbasics and factorizations for CUTEr problems 110
Table 8.6: Number of superbasics and factorizations for Maros and Mészáros problems . 113
Table 8.7: Number of temporarily fixed variables for a vertex for CUTEr problems 116
Table 8.8: Number of temporarily fixed variables for a vertex for Maros and Mészáros problems 117

Table A.1: Problem sizes for CUTEr QPs 118
Table A.2: Problem sizes for Maros and Mészáros QPs 119

LIST OF ALGORITHMS

Algorithm 2.1: Binding-direction method for general QP 19
Algorithm 2.2: Nonbinding-direction method for general QP 27

Algorithm 3.1: Nonbinding-direction method for a general QP in standard form 40

Algorithm 4.1: Dual nonbinding-direction method for a convex QP in standard form . . 54

Algorithm 6.1: Constraint-shifted algorithm 73
Algorithm 6.2: Variable-shifted algorithm 78
Algorithm 6.3: Bound-constrained Lagrangian algorithm 80

ACKNOWLEDGEMENTS

It would several pages to thank my advisor Philip Gill. I am most grateful for his patience and encouragement over the years. I am also thankful for the funding he has provided, and for the opportunities to travel to various conferences and workshops around the world. On a lighter note, I would also like to thank Philip for teaching me the importance of booking air travel in advance, and the aesthetic value of dropping subscripts in LaTeX.

I would like to thank Henry Abarbanel, Randy Bank, Michael Holst, and Alison Marsden for taking time to be on my committee.

I would also like to thank Michael Saunders for numerous conversations about quadratic programming and all things in life, and for his company at several workshops and conferences. I would also like to thank Michael for inviting me to give a talk at Stanford University.

Some of the work in this dissertation would not be possible without the help of Iain Duff, who provided an additional subroutine for HSL_MA57. I also thank Chris Maes for discussions on the interfaces to various testing environments.

Finally, I would like to thank my mom (who jokingly tells people who mention how hard it must be to get a Ph.D., that she doesn't think it's hard because she never sees me working), my sister, and all of my friends, some of whom I don't see for months, yet always seem to be there for me. I thank all of them for their love and support through the years.

VITA

2005	B.A. Computer Science, University of California, San Diego B.A. Mathematics, University of California, San Diego
2007	M.A. Applied Mathematics, University of California, San Diego
2011	Ph.D. Mathematics, University of California, San Diego

PUBLICATIONS

H. D. I. Abarbanel, P. Bryant, P. E. Gill, M. Kostuk, J. Rofeh, Z. Singer, B. Toth, and E. Wong. Dynamical parameter and state estimation in neuron models. In M. Ding and D. Glanzman, editors, *An Exploration of Neuronal Variability and its Functional Significance*, 139–180. Oxford University Press, New York and Oxford, 2011.

P. E. Gill and E. Wong. Sequential quadratic programming methods. In J. Lee and S. Leyffer, editors, *Mixed-Integer Nonlinear Optimization: Algorithmic Advances and Applications*, The IMA Volumes in Mathematics and its Applications. Springer Verlag, Berlin, Heidelberg, and New York, 2011 (To appear).

ABSTRACT OF THE DISSERTATION

Active-Set Methods for Quadratic Programming

by

Elizabeth Wong

Doctor of Philosophy in Mathematics

University of California, San Diego, 2011

Professor Philip E. Gill, Chair

Computational methods are considered for finding a point satisfying the second-order necessary conditions for a general (possibly nonconvex) quadratic program (QP). A framework for the formulation and analysis of feasible-point active-set methods is proposed for a generic QP. This framework is defined by reformulating and extending an inertia-controlling method for general QP that was first proposed by Fletcher and subsequently modified by Gould. This reformulation defines a class of methods in which a primal-dual search pair is the solution of a "KKT system" of equations associated with an equality-constrained QP subproblem defined in terms of a "working set" of linearly independent constraints. It is shown that, under certain circumstances, the solution of this KKT system may be updated using a simple recurrence relation, thereby giving a significant reduction in the number of systems that need to be solved. The use of inertia control guarantees that the KKT systems remain nonsingular throughout, thereby allowing the utilization of third-party linear algebra software.

The algorithm is suitable for indefinite problems, making it an ideal QP solver for stand-alone applications and for use within a sequential quadratic programming method using exact

second derivatives. The proposed framework is applied to primal and dual quadratic problems, as well as to single-phase problems that combine the feasibility and optimality phases of the active-set method, producing a range of formats that are suitable for a variety of applications.

The algorithm is implemented in the Fortran code icQP. Its performance is evaluated using different symmetric and unsymmetric linear solvers on a set of convex and nonconvex problems. Results are presented that compare the performance of icQP with the convex QP solver SQOPT on a large set of convex problems.

1 Introduction

1.1 Overview

Quadratic programming (QP) minimizes a quadratic objective function subject to linear constraints on the variables. A general form of the problem may be written with mixed (equality and inequality) constraints as

$$\begin{array}{ll} \underset{x \in \mathbb{R}^n}{\text{minimize}} & \varphi(x) = c^T x + \tfrac{1}{2} x^T H x \\ \text{subject to} & Ax = b, \quad \ell \leq Dx \leq u, \end{array}$$

where φ is the quadratic objective function, H is the symmetric $n \times n$ Hessian matrix, $c \in \mathbb{R}^n$ is the constant objective vector, A is the $m \times n$ equality constraint matrix, D is the $m_D \times n$ inequality constraint matrix, and ℓ and u are vectors such that $\ell \leq u$.

The difficulty of solving a QP depends on the convexity of the quadratic objective function. If the Hessian matrix H is positive semidefinite, then the QP is convex. In this case, a local solution of the QP is also a global solution. However, when H is indefinite, the QP is nonconvex and the problem is NP-hard—even for the calculation of a local minimizer [12, 32].

The majority of methods for solving quadratic programs can be categorized into either *active-set methods* (which are discussed heavily in Section 2.2) or *interior methods*. Briefly, active-set methods are iterative methods that solve a sequence of equality-constrained quadratic subproblems. The goal of the method is to predict the active set, the set of constraints that are satisfied with equality, at the solution of the problem. The conventional active-set method is divided into two phases; the first focuses on feasibility, while the second focuses on optimality. An advantage of active-set methods is that the methods are well-suited for "warm starts", where a good estimate of the optimal active set is used to start the algorithm. This is particularly useful in applications where a sequence of quadratic programs is solved, e.g., in a sequential quadratic programming method (discussed in the next section) or in an ODE- or PDE-constrained problem with mesh refinements (e.g., see SNCTRL [46], an optimal control interface for nonlinear solver SNOPT). Other applications of quadratic programming include portfolio analysis, structural analysis, and optimal control. Some existing active-set quadratic programming solvers include QPOPT [37], SQOPT [39], and QPA (part of the GALAHAD library) [51].

Interior-point methods compute iterates that lie in the interior of the feasible region, rather than on the boundary of the feasible region. The method computes and follows a continuous path to the optimal solution. In the simplest case, the path is parameterized by a positive scalar that may be interpreted as a perturbation of the optimality conditions for the problem. This parameter also serves as a regularization parameter of the linear equations that are solved at each iteration.

Generally, interior methods require fewer iterations than active-set methods. However, each iteration of interior-point methods is more expensive because the method must solve linear systems involving all the variables of the problem whereas active-set methods solve systems involving some subset of the variables. An advantage of having all variables in the equations makes the dimension of the equations and the sparsity pattern of the matrix involved fixed throughout. The path-following feature of interior-point methods also causes difficulties when the problem is warm-started, as a warm-start point is typically far from the path and many iterations are required to move onto the path. IPOPT [68] and LOQO [67] are two examples of interior-point codes.

To simplify the exposition, inequality constraints with only lower bounds are considered in this thesis, although the methods are easily extended to problems with lower and upper bounds. The simplified mixed-constraint QP becomes

$$\begin{array}{ll} \underset{x\in\mathbb{R}^n}{\text{minimize}} & \varphi(x) = c^Tx + \tfrac{1}{2}x^THx \\ \text{subject to} & Ax = b, \quad Dx \geq f, \end{array} \tag{1.1}$$

where f is a constant vector. If the inequality constraint matrix D is the identity matrix, and the vector f of lower bounds is zero, then the problem said to be in *standard-form*, where the constraints are linear equalities and simple bounds on the variables:

$$\begin{array}{ll} \underset{x\in\mathbb{R}^n}{\text{minimize}} & \varphi(x) = c^Tx + \tfrac{1}{2}x^THx \\ \text{subject to} & Ax = b, \quad x \geq 0. \end{array} \tag{1.2}$$

Every QP may be written in standard form. For example, consider a problem with a mixture of general inequalities and simple bounds:

$$\underset{x\in\mathbb{R}^n, s\in\mathbb{R}^m}{\text{minimize}}\ \varphi(x) = c^Tx + \tfrac{1}{2}x^THx \ \text{ subject to }\ Ax \geq 0, \quad x \geq 0.$$

By introducing a set of nonnegative slack variables s, the all-inequality problem may be rewritten as

$$\begin{array}{ll} \underset{x\in\mathbb{R}^n, s\in\mathbb{R}^m}{\text{minimize}} & \varphi(x) = c^Tx + \tfrac{1}{2}x^THx \\ \text{subject to} & Ax - s = 0, \quad x \geq 0, \quad s \geq 0. \end{array} \tag{1.3}$$

The advantage of including slack variables is that the constraint matrix $\begin{pmatrix} A & -I \end{pmatrix}$ trivially has full row rank, which is an important assumption in the methods to be described. However, for simplicity, we do not include slack variables explicitly our discussion, but consider only problems of the form (1.1) or (1.2) with the assumption that the constraint matrix A has full row rank.

1.2 Contributions of this Thesis

Our work in quadratic programming is driven by our interest in *nonlinear programming* (NLP), the minimization of nonlinear functions subject to nonlinear constraints. An important algorithm for NLP is *sequential quadratic programming* (or SQP). The method solves a sequence of quadratic subproblems whose objective function is a quadratic model of the nonlinear objective subject to a linearization of the constraints.

The purpose of the work in this thesis is to address some of the difficulties that arise in SQP methods. In general, it is difficult to implement SQP methods using exact second derivatives because the QP subproblems can be nonconvex. The complexity of the QP subproblem has been a major impediment to the formulation of second-derivative SQP methods (although methods based on indefinite QP have been proposed by Fletcher and Leyffer [30, 31]). To avoid this difficulty, algorithm developers refrain from using exact second derivatives and instead use a positive-semidefinite approximation of the Hessian to define convex QP subproblems (see SNOPT [38]). Another difficulty associated with conventional SQP methods is the reliance on customized linear algebra software. This prevents algorithms from taking advantage of advanced linear algebra solvers that can exploit developments in computer hardware and architecture. (For a detailed review of SQP methods, see [44].) In addition, the presented algorithm will address some of the deficiencies of the existing convex QP solver SQOPT [39]. The goal is for this work to complement the capabilities of SQOPT, in order to cover a larger range of problems and applications.

A framework for the formulation and analysis of feasible-point active-set methods is proposed for a generic QP. This framework is discussed in the context of two broad classes of active-set method for quadratic programming: *binding-direction methods* and *nonbinding-direction methods*. Broadly speaking, the working set for a binding-direction method consists of a subset of the active constraints, whereas the working set for a nonbinding direction method may involve constraints that need not be active (nor even feasible). A binding-direction method for general QP, first proposed by Fletcher [29] and subsequently modified by Gould [49], is recast as a nonbinding-direction method. This reformulation defines a class of methods in which a primal-dual search pair is the solution of a "KKT system" of equations associated with an equality-constrained QP subproblem defined in terms of a "working set" of linearly independent constraints. It is shown that, under certain circumstances, the solution of this KKT system may be updated using a simple recurrence relation, thereby giving a significant reduction in the number of systems that need to be solved. This framework addresses the current difficulties of QP methods, creating an algorithm that is suitable for indefinite problems and that is capable of utilizing external linear algebra software.

In Chapter 2, we provide background information on active-set methods. Detailed descriptions of the binding-direction and nonbinding-direction methods are also given for problems in mixed-format, providing the framework for the methods discussed in subsequent chapters. In

Chapter 3, the nonbinding-direction method is defined for standard-form problems. It will be shown that the standard-form version of the algorithm leads to a reduction in the dimension of the KKT systems solved at each iteration. This form of the nonbinding-direction method is implemented in the Fortran code icQP, and numerical results of this implementation are discussed in Chapter 8. Chapter 4 considers the application of the nonbinding-direction method to the dual of a convex quadratic program. Many existing dual methods require the inverse of the Hessian, limiting the methods to strictly convex problems. It will be shown that the method presented is appropriate for problems that are not strictly convex. Chapter 5 addresses the issues of computing an initial point for the algorithm. In Chapter 6, single-phase methods that combine the feasibility and optimality phases of the active-set method are described. Two methods involving variants of the *augmented Lagrangian function* are derived. Chapter 7 describes the two methods for solving the linear equations involved in the QP method. The first approach utilizes a symmetric transformation of the reduced Hessian matrix. The second approach uses a symmetric indefinite factorization of a fixed KKT matrix with the factorization of a smaller matrix that is updated at each iteration of the method.

1.3 Notation, Definitions, and Useful Results

The vector $g(x)$ denotes $c + Hx$, the gradient of the objective φ evaluated at x. Occasionally, the gradient will be referred to as simply g. The vector d_i^T refers to the i-th row of the constraint matrix D, so that the i-th inequality constraint is $d_i^T x \geq f_i$. The i-th component of a vector labeled with a subscript will be denoted by $[\,\cdot\,]_i$, e.g., $[v_N]_i$ is the i-th component of the vector v_N. Similarly, a subvector of components with indices in the index set $\mathcal{S}$ is denoted by $(\,\cdot\,)_{\mathcal{S}}$, e.g., $(v_N)_{\mathcal{S}}$ is the vector with components $[v_N]_i$ for $i \in \mathcal{S}$. The symbol I is used to denote an identity matrix with dimension determined by the context. The j-th column of I is denoted by e_j. The vector e will be used to denote the vector of all ones with length determined by the context. The vector with components $\max\{-x_i, 0\}$ (i.e., the magnitude of the negative part of x) is denoted by $[\,x\,]_-$. Unless explicitly indicated otherwise, $\|\cdot\|$ denotes the vector two-norm or its induced matrix norm. Given vectors a and b with the same dimension, the vector with i-th component $a_i b_i$ is denoted by $a \cdot b$. For any set $\mathcal{S}$ and index s, the notation $\mathcal{S} + \{s\}$ is used to denote the addition of s to the set $\mathcal{S}$. Similar notation with the symbol "$-$" is used to denote the removal of an index. Given vectors x, y and z, the long vector consisting of the elements of x augmented by elements of y and z is denoted by (x, y, z).

Definition 1.3.1 (Inertia of a matrix)**.** *Given a symmetric matrix A, its* inertia*, denoted by* $\mathrm{In}(A)$ *is the integer triple* (a_+, a_-, a_0)*, giving the number of positive, negative and zero eigenvalues of A.*

Result 1.3.1 (Sylvester's Law of Inertia)**.** *Given a symmetric matrix A and a nonsingular matrix*

S, then $\text{In}(S^TAS) = \text{In}(A)$.

Theorem 1.3.1. *Given an $n \times n$ symmetric matrix H and an $m \times n$ matrix A, let r denote the rank of A and let the columns of Z form a basis for the null space of A. If K is the matrix*

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

Corollary 1.3.1. *Given an $n \times n$ symmetric matrix H and an $m \times n$ matrix A of rank m, let the columns of Z form a basis for the null space of A. If K is the matrix*

$$K = \begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix},$$

then $\text{In}(K) = \text{In}(Z^THZ) + (m, m, 0)$*. If Z^THZ is positive definite, then* $\text{In}(K) = (n, m, 0)$ *and we say that K has* correct inertia*.*

Theorem 1.3.2. *Let H be an $n \times n$ symmetric matrix, A be an $m \times n$ matrix and scalar $\mu > 0$. Define K as the matrix*

$$K = \begin{pmatrix} H & A^T \\ A & -\mu I \end{pmatrix}.$$

Then $\text{In}(K) = \text{In}(H + \dfrac{1}{\mu}A^TA) + \text{In}(0, m, 0)$.

Proof. Define the nonsingular matrix S

$$S = \begin{pmatrix} I & 0 \\ \frac{1}{\mu}A & I \end{pmatrix}.$$

By Sylvester's Law of Inertia,

$$\begin{aligned} \text{In}(K) = \text{In}(S^TKS) &= \text{In} \begin{pmatrix} H + \frac{1}{\mu}A^TA & 0 \\ 0 & -\mu I \end{pmatrix} \\ &= \text{In}(H + \frac{1}{\mu}A^TA) + (0, m, 0). \end{aligned}$$

□

Result 1.3.2 (Debreu's Lemma)**.** *Given an $m \times n$ matrix A and an $n \times n$ symmetric matrix H, then $x^THx > 0$ for all nonzero x satisfying $Ax = 0$ if and only if there is a finite $\bar{\mu} \geq 0$ such that $H + \frac{1}{\mu}A^TA$ is positive definite for all $0 < \mu \leq \bar{\mu}$.*

Result 1.3.3 (Schur complement)**.** *Given a symmetric matrix*

$$K = \begin{pmatrix} M & N^T \\ N & G \end{pmatrix}, \tag{1.4}$$

with M nonsingular, the Schur complement *of M in K will be denoted by K/M, and is defined as*

$$K/M \triangleq G - NM^{-1}N^T.$$

Moreover, $\mathrm{In}(K) = \mathrm{In}(K/M) + \mathrm{In}(M)$*. We sometimes refer simply to "the" Schur complement when the relevant matrices are clear.*

Result 1.3.4 (Symmetric indefinite factorization)**.** *Let K be an $n \times n$ symmetric matrix with rank r. Then there exists a permutation matrix P, a unit upper-triangular matrix U, and a block diagonal matrix D such that*

$$P^TKP = U^TDU, \quad \textit{with} \quad D = \mathrm{diag}(D_1, D_2, \ldots, D_s, 0_{n-r,n-r}),$$

where each D_j is nonsingular and has dimension 1×1 or 2×2. Moreover, each of the 2×2 blocks has one positive and one negative eigenvalue. The equivalent factorization

$$K = LDL^T, \quad \textit{with} \quad L = (PU)^T,$$

is known as the LDL^T factorization.

Lemma 1.3.1. *Let A be a $m \times n$ matrix of full row rank* ($\mathrm{rank}(A) = m$) *and g be any n-vector.*

(a) *If $g = A^Ty$ and there exists an index s such that $y_s < 0$, then there exists a vector p such that $g^Tp < 0$ and $Ap \geq 0$.*

(b) *$g \notin \mathrm{range}(A^T)$ if and only if there exists a vector p such that $g^Tp < 0$ and $Ap = 0$.*

Result 1.3.5 (The interlacing eigenvalue property)**.** *Assume K is a symmetric $n \times n$ matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Suppose that K is partitioned so that*

$$K = \begin{pmatrix} A & B \\ B^T & C \end{pmatrix}$$

with A $m \times m$. If the eigenvalues of A are $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_m$, then

$$\lambda_{k+n-m} \leq \mu_k \leq \lambda_k, \quad k = 1, 2, \ldots, m.$$

2 Quadratic Programming

This chapter introduces the framework for the formulation and analysis of active-set methods for quadratic programs. The framework is described for problems in mixed format, which involve minimizing a quadratic objective function subject to linear equality and inequality constraints. The problem is assumed to be of the form

$$\begin{array}{ll} \underset{x\in\mathbb{R}^n}{\text{minimize}} & \varphi(x) = c^Tx + \frac{1}{2}x^THx \\ \text{subject to} & Ax = b, \quad Dx \geq f, \end{array} \tag{2.1}$$

where $\varphi(x) : \mathbb{R}^n \to \mathbb{R}$ is the quadratic objective function, the Hessian matrix H is symmetric and the constraint matrices A and D are $m \times n$ and $m_D \times n$, respectively. Without loss of generality, A is assumed to have rank m. No assumptions are made about H other than symmetry, which implies that the objective φ need not be convex. In the nonconvex case, however, convergence will be to local minimizers only.

Section 2.1 provides information on the optimality conditions of mixed-constraint problems. Section 2.2 introduces a general class of methods for solving QPs known as active-set methods. In Sections 2.2.1 and 2.2.2, two particular active-set method based on inertia control are presented. The remaining sections extend the method to quadratic programs in different formats, and discuss the relationship of the method to the simplex method for linear programs.

2.1 Background

The necessary and sufficient conditions for a local solution of the QP (2.1) involve the existence of vectors π and z of Lagrange multipliers associated with the constraints $Ax = b$ and $Dx \geq f$, respectively.

Definition 2.1.1 (First-order KKT point). *A point x is a* first-order KKT point *for* (2.1) *if* *there exists at least one pair of Lagrange multiplier vectors* π *and* z *such that*

$$\begin{aligned} Ax = b, \quad Dx \geq f & \qquad \textit{(feasibility)} \\ g(x) = A^T\pi + D^T z & \qquad \textit{(stationarity)} \\ z \geq 0 & \qquad \textit{(nonnegativity)} \\ z \cdot (Dx - f) = 0 & \qquad \textit{(complementarity).} \end{aligned}$$

Following conventional linear programming terminology, the x variables are referred to as the *primal variables* and the Lagrange multipliers π and z are the *dual variables.* We may refer to a first-order KKT point x together with its Lagrange multipliers as (x, π, z).

In addition to being a first-order KKT point, a point x must also satisfy certain second-order conditions to be a local solution of the QP. The conditions are summarized by the following result, which is stated without proof (see, e.g., Borwein [7], Contesse [12] and Majthay [55]).

Result 2.1.1 (QP optimality conditions)**.** *The point* x^* *is a local minimizer of the quadratic program* (2.1) *if and only if*

(a) x^* *is a first-order KKT point, and*

(b) $p^T H p \geq 0$ *for all nonzero* p *satisfying* $g(x)^T p = 0$, $Ap = 0$, *and* $d_i^T p \geq 0$ *for every* i *such that* $d_i^T x^* = f_i$.

If H has at least one negative eigenvalue and (x, π, z) is a first-order KKT point with an index i such that $z_i = 0$ and $d_i^T x = f_i$, then x is known as a *dead point.* Verifying condition (b) at a dead point requires finding the global minimizer of an indefinite quadratic form over a cone, which is an NP-hard problem (see, e.g., Cottle, Habetler and Lemke [14], Pardalos and Schnitger [58], and Pardalos and Vavasis [59]). This implies that the optimality of a candidate solution of a general quadratic program can be verified only if more restrictive (but computationally tractable) sufficient conditions are satisfied. A dead point is a point at which the sufficient conditions are not satisfied, but certain necessary conditions for optimality hold. Computationally tractable necessary conditions are based on the following result.

Result 2.1.2 (Necessary conditions for optimality)**.** *The point* x^* *is a local minimizer of the QP* (2.1) *only if*

(a) x^* *is a first-order KKT point;*

(b) *it holds that* $p^T H p \geq 0$ *for all nonzero* p *satisfying* $Ap = 0$, *and* $d_i^T p = 0$ *for each* i *such that* $d_i^T x^* = f_i$.

Suitable sufficient conditions for optimality are given by (a)–(b) with (b) replaced by the condition that $p^T H p \geq 0$ for all p such that $Ap = 0$, and $d_i^T p = 0$ for every $i \in \mathcal{A}_+(x^*)$, where $\mathcal{A}_+(x^*)$ is the index set $\mathcal{A}_+(x^*) = \{i : d_i^T x^* = f_i \text{ and } z_i > 0\}$.

These conditions may be expressed in terms of the constraints that are satisfied with equality at x^*. Let x be any point satisfying the equality constraints $Ax = b$. (The assumption that A has rank m implies that there must exist at least one such x.) An inequality constraint is *active* at x if it is satisfied with equality. The indices associated with the active constraints comprise the *active set*, denoted by $\mathcal{A}(x)$. An *active-constraint matrix* $A_{\mathfrak{a}}(x)$ is a matrix with rows consisting of the rows of A and the gradients of the active constraints. By convention, the rows of A are listed first, giving the active-constraint matrix

$$A_{\mathfrak{a}}(x) = \begin{pmatrix} A \\ D_{\mathfrak{a}}(x) \end{pmatrix},$$

where $D_{\mathfrak{a}}(x)$ comprises the rows of D with indices in $\mathcal{A}(x)$. Let $m_{\mathfrak{a}}$ denote the number of indices in $\mathcal{A}(x)$, so that the number of rows in $A_{\mathfrak{a}}(x)$ is $m + m_{\mathfrak{a}}$. The argument x is generally omitted if it is clear where $D_{\mathfrak{a}}$ is defined.

With this definition of the active set, an equivalent statement of Result 2.1.2 is given.

Result 2.1.3 (Necessary conditions in active-set form)**.** *Let the columns of the matrix $Z_{\mathfrak{a}}$ form a basis for the null space of $A_{\mathfrak{a}}$. The point x^* is a local minimizer of the QP* (2.1) *only if*

(a) *x^* is a first-order KKT point, i.e.,* (i) $Ax^* = b$, $Dx^* \geq f$*;* (ii) *$g(x^*)$ lies in* $\operatorname{range}(A_{\mathfrak{a}}^T)$, *or equivalently, there exist vectors π^* and $z_{\mathfrak{a}}^*$ such that $g(x^*) = A^T\pi^* + D_{\mathfrak{a}}^T z_{\mathfrak{a}}^*$; and* (iii) $z_{\mathfrak{a}}^* \geq 0$,

(b) *the reduced Hessian $Z_{\mathfrak{a}}^T H Z_{\mathfrak{a}}$ is positive semidefinite.*

2.2 Active-Set Methods for Mixed-Constraint Problems

Active-set methods are two-phase iterative methods that provide an estimate of the active set at the solution. In the first phase (the *feasibility phase* or *phase 1*), the objective is ignored while a feasible point is found for the constraints $Ax = b$ and $Dx \geq f$. In the second phase (the *optimality phase* or *phase 2*), the objective is minimized while feasibility is maintained. For efficiency, it is beneficial if the computations of both phases are performed by the same underlying method. The two-phase nature of the algorithm is reflected by changing the function being minimized from a function that reflects the degree of infeasibility to the quadratic objective function. For this reason, it is helpful to consider methods for the optimality phase first. Methods for the feasibility phase are considered in Chapter 5.

Given a feasible point x_0, active-set methods compute a sequence of feasible iterates $\{x_k\}$ such that $x_{k+1} = x_k + \alpha_k p_k$ and $\varphi(x_{k+1}) \leq \varphi(x_k)$, where p_k is a nonzero search direction and α_k is a nonnegative step length. Active-set methods are motivated by the main result of Farkas' Lemma, which states that a feasible x must either satisfy the first-order optimality conditions or be the starting point of a *feasible descent direction*, i.e., a direction p such that

$$A_{\mathfrak{a}}\, p \geq 0 \quad \text{and} \quad g(x)^T p < 0. \tag{2.2}$$

In most of the active-set methods considered here, the active set is approximated by a *working set* $\mathcal{W}$ of row indices of D. The working set has the form $\mathcal{W} = \{\nu_1, \nu_2, \ldots, \nu_{m_w}\}$, where m_w is the number of indices in $\mathcal{W}$. Analogous to the active-constraint matrix $A_{\mathfrak{a}}$, the $(m+m_w)\times n$ *working-set matrix* A_w contains the gradients of the equality constraints and inequality constraints in $\mathcal{W}$. The structure of the working-set matrix is similar to that of the active-set matrix, i.e.,

$$A_w = \begin{pmatrix} A \\ D_w \end{pmatrix},$$

where D_w is a matrix formed from the m_w rows of D with indices in $\mathcal{W}$. The vector f_w denotes the components of f with indices in $\mathcal{W}$.

There are two important distinctions between the definitions of $\mathcal{A}$ and $\mathcal{W}$.

(a) The indices of $\mathcal{W}$ must define a subset of the rows of D that are linearly independent of the rows of A, i.e., the working set matrix A_w has full row rank. It follows that m_w must satisfy $0 \leq m_w \leq \min\{n-m, m_D\}$.

(b) The active set $\mathcal{A}$ is uniquely defined at any feasible x, whereas there may be many choices for $\mathcal{W}$. The set $\mathcal{W}$ is determined by the properties of the particular active-set method being employed.

Conventional active-set methods define the working set as a subset of the active set (as in the method described in Section 2.2.1). However, in the considered method of Section 2.2.2, the requirement is relaxed—a working-set constraint need not necessarily be active at x.

Given a working set $\mathcal{W}$ and an associated working-set matrix A_w at x, the notions of stationarity and optimality with respect to $\mathcal{W}$ are introduced.

Definition 2.2.1 (Subspace stationary point)**.** *Let $\mathcal{W}$ be a working set defined at x such that $Ax = b$. Then x is a* subspace stationary point with respect to $\mathcal{W}$ (or, equivalently, with respect to A_w) *if $g(x) \in \operatorname{range}(A_w^T)$, i.e., there exists a vector y such that $g(x) = A_w^T y$. Equivalently, x is a subspace stationary point with respect to the working set $\mathcal{W}$ if the reduced gradient $Z_w^T g(x)$ is zero, where the columns of Z_w form a basis for the null space of A_w.*

At a subspace stationary point, the components of y are the Lagrange multipliers associated with a QP with equality constraints $Ax = b$ and $D_w x = f_w$. To be consistent with the optimality conditions of Result 2.1.3, the first m components of y are denoted as π (the multipliers associated with $Ax = b$) and the last m_w components of y as z_w (the multipliers associated with the constraints in $\mathcal{W}$). With this notation, the identity $g(x) = A_w^T y = A^T\pi + D_w^T z_w$ holds at a subspace stationary point.

To classify subspace stationary points based on curvature information, we define the terms *second-order-consistent working set* and *subspace minimizer.*

Definition 2.2.2 (Second-order-consistent working set)**.** *Let $\mathcal{W}$ be a working set associated with x such that $Ax = b$, and let the columns of Z_w form a basis for the null space of A_w. The working set $\mathcal{W}$ is* second-order consistent *if the reduced Hessian $Z_w^T H Z_w$ is positive definite.*

The inertia of the reduced Hessian is related to the inertia of the $(n+m+m_w) \times (n+m+m_w)$ KKT matrix $K = \begin{pmatrix} H & A_w^T \\ A_w & \end{pmatrix}$ through the identity $\mathrm{In}(K) = \mathrm{In}(Z_w^T H Z_w) + (m+m_w, m+m_w, 0)$ from Theorem 1.3.1. It follows that an equivalent characterization of a second-order-consistent working set is that K has inertia $(n, m+m_w, 0)$, in which case, K is said to have *correct inertia.*

Definition 2.2.3 (Subspace minimizer)**.** *If x is a subspace stationary point with respect to a second-order-consistent basis $\mathcal{W}$, then x is known as a* subspace minimizer with respect to $\mathcal{W}$. *If every constraint in the working set is active, then x is called a* standard subspace minimizer*; otherwise x is called a* nonstandard subspace minimizer.

A *vertex* is a point at which $\mathrm{rank}(A_{\mathfrak{a}}) = n$ and $m_{\mathfrak{a}} \geq n - m$. If $\mathrm{rank}(A_{\mathfrak{a}}) = n$, then the null space of $A_{\mathfrak{a}}$ is trivial, so that a vertex such that $g(x) \in \mathrm{range}(A_{\mathfrak{a}}^T)$ is a subspace minimizer. A feasible x is said to be a *degenerate point* if $g(x)$ lies in $\mathrm{range}(A_{\mathfrak{a}}^T)$ and the rows of $A_{\mathfrak{a}}$ are linearly dependent, i.e., $\mathrm{rank}(A_{\mathfrak{a}}) < m + m_{\mathfrak{a}}$. If exactly $n - m$ constraints of $Dx \geq f$ are active at a vertex, then the vertex is *nondegenerate.* If more than $n - m$ are active, then the vertex is *degenerate.* At a degenerate point there are infinitely many vectors y such that $g(x) = A_{\mathfrak{a}}^T y$. Moreover, at least one of these vectors has a zero component. Degeneracy can be a problem as it can lead to dead points. Degenerate points can also lead to *cycling*, where the active-set method does not move from the current iterate but returns to an earlier working set, causing an infinite sequence where the same working sets are repeated.

In the following sections, two active-set methods for solving QPs are described, the *binding-direction method* and the *nonbinding-direction method.* In the binding-direction method, every direction lies in the null space of the working-set matrix, so that all working-set constraints are active or *binding.* In the nonbinding-direction method, directions are nonbinding (inactive) with respect to one of the constraints in the working set. Both methods produce the same sequence of iterates and differ only in the equations solved at each step. The binding-direction method is tied to a specific method for modifying the factors of the working-set matrix. The nonbinding-direction method is designed so that only nonsingular systems are solved at each step, making the method more easily adapted for use with general-purpose solvers. Both methods are inertia-controlling methods that limit the number of nonpositive eigenvalues in the KKT matrices. In the binding-direction method, the reduced Hessians are limited to having at most one non-positive eigenvalue, while the nonbinding-direction method computes only subspace minimizers (e.g., working sets that define positive-definite reduced Hessians) at each iteration.

2.2.1 Binding-direction method

The binding-direction produces a sequence of iterates that begins and ends at a subspace minimizer but defines intermediate iterates that are not subspace minimizers. One iteration of the method is described. The working-set matrix A_w at the k-th iteration will be denoted by A_k to differentiate between the working sets at different iterates. Similar notation for other vectors or matrices with working-set subscripts apply.

The method starts at a standard subspace minimizer x_k with working set $\mathcal{W}_k$, i.e., $g_k = A_k^T y_k$ for a unique y_k and a reduced Hessian matrix $Z_k^T H Z_k$ that is positive definite. If x_k is non-optimal, then there exists an index $\nu_s \in \mathcal{W}_k$ such that $[y_k]_{m+s} < 0$. By part (i) of Lemma 1.3.1, there exists a descent direction for φ such that $g_k^T p < 0$ and $A_k p = e_{m+s}$. Instead of imposing the condition that $A_k p = e_{m+s}$, we increase the iteration counter to $k+1$ and set $x_k = x_{k-1}$. The new working set is defined as $\mathcal{W}_k = \mathcal{W}_{k-1} - \{\nu_s\}$, and y_k be the vector y_{k-1} with the $(m+s)$-th component removed. The removal of $d_{\nu_s}^T x \geq f_{\nu_s}$ means that x_k is no longer a subspace stationary point with respect to $\mathcal{W}_k$ since

$$g(x_k) = g(x_{k-1}) = A_{k-1}^T y_{k-1} = A_k^T y_k + [y_{k-1}]_{m+s} d_{\nu_s} \;\text{ with }\; [y_{k-1}]_{m+s} < 0, \tag{2.3}$$

and hence $g(x_k) \notin \operatorname{range}(A_k^T)$. In this case, there exists a descent direction in the null space of A_k such that

$$g_k^T p < 0 \;\text{ and }\; A_k p = 0, \;\text{ and }\; d_{\nu_s}^T p > 0. \tag{2.4}$$

The direction p is a *binding* direction because the constraints in the working set remain active for any step along p. The first two conditions of (2.4) are satisfied by part (ii) of Lemma 1.3.1. For the last condition, first note that $x_k = x_{k-1}$, $g_k = g_{k-1} = A_{k-1}^T y_{k-1}$ with $[y_{k-1}]_{m+s} < 0$ and the working-set matrix A_k is A_{k-1} with the constraint normal $d_{\nu_s}^T$ removed. The identity $A_k p_k = 0$ implies that p_k must be orthogonal to every row of A_{k-1} except $d_{\nu_s}^T$. Thus,

$$\begin{aligned} 0 > g_k^T p_k = g_{k-1}^T p_k &= p_k^T (A_{k-1}^T y_{k-1}) \\ &= (d_{\nu_s}^T p_k) e_{m+s}^T y_{k-1} = (d_{\nu_s}^T p_k)[y_{k-1}]_{m+s}. \end{aligned}$$

It follows that $d_{\nu_s}^T p_k > 0$ and hence p_k satisfies (2.4).

An obvious choice for p_k is the solution of the equality-constrained quadratic program

$$\underset{p}{\text{minimize}} \;\; \varphi(x_k + p) \quad \text{subject to} \quad A_k p = 0. \tag{2.5}$$

Assume for the moment that this problem has a bounded solution (i.e., that $Z_k^T H Z_k$ is positive definite). The optimality conditions for (2.5) imply the existence of vector q_k such that $g(x_k + p_k) = A_k^T (y_k + q_k)$, i.e., q_k defines the step to the multipliers at the optimal solution $x_k + p_k$. This optimality condition combined with the feasibility condition imply that p_k and q_k satisfy the KKT equations

$$\begin{pmatrix} H & A_k^T \\ A_k & 0 \end{pmatrix} \begin{pmatrix} p_k \\ -q_k \end{pmatrix} = - \begin{pmatrix} g_k - A_k^T y_k \\ 0 \end{pmatrix}. \tag{2.6}$$

The point $x_k + p_k$ is a subspace minimizer with respect to $\mathcal{W}$, with appropriate multiplier vector $y_k + q_k$. If the KKT matrix is indefinite (but not singular), a direction is still computed from (2.6), though $x_k + p_k$ will not be a subspace minimizer and p_k must be limited by some step (discussed in the next subsection).

For any scalar α, the direction defined in (2.6) satisfies

$$\begin{aligned} g(x_k + \alpha p_k) = g_k + \alpha H p_k &= g_k + \alpha(-(g_k - A_k^T y_k) + A_k^T q_k) \\ &= (1-\alpha) g_k + \alpha A_k^T (y_k + q_k) \\ &= (1-\alpha)(A_k^T y_k + y_{m+s} d_{\nu_s}) + \alpha A_k^T (y_k + q_k) \\ &= (1-\alpha) y_{m+s} d_{\nu_s} + A_k^T (y_k + \alpha q_k), \end{aligned} \tag{2.7}$$

using the identity in (2.3).

If the KKT matrix of (2.6) is singular, or equivalently, the associated reduced Hessian $Z_k^T H Z_k$ is singular, the subproblem (2.5) is unbounded and the system (2.6) cannot be used to define p_k. In this situation, a direction is found such that

$$g_k^T p_k < 0, \quad p_k^T H p_k = 0 \text{ and } A_k p_k = 0.$$

This vector, called a *descent direction of zero curvature*, is a descent direction such that $Hp_k = 0$. Since the KKT matrix is singular, it must have a null vector, and p_k and q_k may be computed from the system

$$\begin{pmatrix} H & A_k^T \\ A_k & 0 \end{pmatrix} \begin{pmatrix} p_k \\ -q_k \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{2.8}$$

In this case, the directions p_k and q_k satisfy

$$g(x_k + \alpha p_k) - A_k^T (y_k + \alpha q_k) = g_k - A_k^T y_k + \alpha(H p_k - A_k^T q_k) = g_k - A_k^T y_k, \tag{2.9}$$

for every scalar α, so that the norm $\|g_k - A_k^T y_k\|$ is unchanged by any step $x_k + \alpha p_k$.

Solving for the direction. Regardless of whether p_k is computed from (2.6) or (2.8), it must be a descent direction, i.e., $g_k^T p_k < 0$. There are three basic approaches to solving either equation (2.6) or (2.8), each of which can utilize direct methods or iterative methods. A *range-space method* requires H to be nonsingular and solves (2.6) by solving the equivalent equations

$$A_k H^{-1} A_k^T q_k = A_k H^{-1} (g_k - A_k^T y_k) \text{ and } Hp = -(g_k - A_k^T y_k) + A_k^T q_k, \tag{2.10}$$

which require a solve with H and a factorization of the matrix $A_k H^{-1} A_k^T$. Obviously, the need for nonsingular H limits this method to strictly convex problems.

The equations (2.6) and (2.8) may also be solved by computing some matrix factorization, e.g., a symmetric indefinite LDL^T factorization of the KKT matrix (see Result 1.3.4). This *full-space method* works directly with the KKT system, but is impractical in an active-set method as the KKT matrix changes at every iteration.

A more appropriate method for computing p_k is a *null-space method*, which computes p_k as $p_k = Z_k p_Z$. If $Z_k^T H Z_k$ is positive definite, then p_Z is the unique solution of

$$Z_k^T H Z_k p_Z = -Z_k^T (g_k - A_k^T y_k),$$

which is an $n \times (n - m - m_k)$ system. If the reduced Hessian is singular, then p_Z may be any vector such that

$$Z_k^T H Z_k p_Z = 0 \text{ and } g_k^T Z_k p_Z < 0.$$

The computation of an LDL^T factorization of a symmetrically permuted reduced Hessian may be used to detect singularity and compute p_Z. When the QP is strictly convex, H is positive definite and at every iterate, the reduced Hessian is positive definite. The matrix can be factored such that

$$Z_k^T H Z_k = R_k^T R_k \text{ and } A_k Q_k = \begin{pmatrix} 0 & T_k \end{pmatrix}, \tag{2.11}$$

where R_k and T_k are upper triangular, and Q_k is an orthogonal matrix that forms an orthogonal basis for the null space of A_k. These factors can then be used to solve the equations in the null-space method above. In addition, the factors may be modified when constraints are added or deleted from the working set. This amounts to significantly less work than it would take to recompute the factorizations from scratch.

If φ is not strictly convex, then $Z_k^T H Z_k$ can have an arbitrary number of nonpositive eigenvalues. It is not possible to modify the factorizations in (2.11) in a way that is efficient and numerically stable. At each iteration, it is necessary to decide which of the two systems should be solved. If the relevant factorizations are computed from scratch at each iteration, then the difficulties can be overcome, though the reliable numerical estimation of rank is a difficult problem. If the factors are modified at each step, then it is much more difficult to compute factors that provide a reliable estimate of the rank. Similar difficulties arise in full-space methods based on direct factorization of the KKT matrix in (2.6) or (2.8).

Each of the three methods above may also utilize iterative methods to solve the linear systems. In particular, when the matrix is positive definite, the conjugate-gradient method can be applied. However, iterative methods may take many iterations to converge to a solution and ill-conditioning may cause difficulties in constructing a preconditioner.

Computing a step length. Once a direction is found, an appropriate step α must be computed. Since p_k is a descent direction, there must exist $\hat{\alpha} > 0$ such that $\varphi(x_k + \alpha p_k) < \varphi(x_k)$ for all $0 < \alpha \le \hat{\alpha}$. If $Z_k^T H Z_k$ is positive definite, p_k is defined by (2.6) and $p_k^T H p_k > 0$, so that φ has positive curvature along p_k. In this case, there is a unique and computable local minimizer α_* of $\varphi(x_k + \alpha p_k)$ with respect to α. As α_* must be a stationary point, it must satisfy

$$\frac{d}{d\alpha} \varphi(x_k + \alpha p_k)\big|_{\alpha=\alpha_*} = g(x_k + \alpha_* p_k)^T p_k = g_k^T p_k + \alpha_* p_k^T H p_k = 0.$$

The unique step α_* from x_k to the local minimizer of φ along the descent direction p_k is given by

$$\alpha_* = -g_k^T p_k / p_k^T H p_k. \tag{2.12}$$

However, the first equation of (2.6) implies that $p_k^T H p_k = -g_k^T p_k$, so that $\alpha_* = 1$.

If $Z_k^T H Z_k$ is indefinite or singular, then no minimizer exists and $\alpha_* = +\infty$. The direction p_k satisfies the identity

$$\varphi(x_k + \alpha p_k) = \varphi(x_k) + \alpha g_k^T p_k + \tfrac{1}{2}\alpha^2 p_k^T H p_k.$$

In particular, when p_k is a direction of zero curvature defined by (2.8), then $\varphi(x_k + \alpha p_k) = \varphi(x_k) + \alpha g_k^T p_k$, which implies that φ is linear along p_k and is unbounded below for $\alpha > 0$. In the indefinite case, φ is unbounded below for $\alpha > 0$ since $p_k^T H p_k < 0$ and $g_k^T p_k < 0$.

If $x_k + \alpha_* p_k$ is infeasible or $\alpha_* = \infty$, then the *maximum feasible step* from x_k along p_k is computed as

$$\alpha_F = \min \gamma_i, \quad \text{with} \quad \gamma_i = \begin{cases} \dfrac{d_i^T x - f_i}{-\, d_i^T p_k} & \text{if } d_i^T p_k < 0, \\ +\infty & \text{otherwise,} \end{cases} \tag{2.13}$$

where any constraint satisfying $d_i^T p_k < 0$ is a decreasing constraint along p_k. The decreasing constraint with index r such that $\alpha_F = \gamma_r$ is called a *blocking constraint*. While there may be several blocking constraints, the value of α_F is unique. Once α_F is computed, the next iterate is defined as $x_{k+1} = x_k + \alpha_k p_k$, where $\alpha_k = \min\{\alpha_*, \alpha_F\}$. If $\alpha_k = +\infty$, then p_k must be a descent direction of zero or negative curvature along which there is no blocking constraint. This means the QP is unbounded and the algorithm terminates. Otherwise, if $\alpha_* \le \alpha_F$, we take an unconstrained step and $x_k + p_k$ is feasible and a subspace minimizer with respect to A_k. If $\alpha_F < \alpha_*$, then the working set is modified to include a blocking constraint that is active at x_{k+1}, e.g., $\mathcal{W}_{k+1} = \mathcal{W}_k + \{r\}$. If multiple blocking constraints exist, only one is chosen to be added. Lemma 1.3.1 implies that any decreasing constraint must be linearly independent of the constraints in the working set.

If x is a degenerate point (a point where the active constraint normals are linearly dependent), then there exists at least one active constraint not in the working set. If this active constraint is decreasing along p_k, then $\alpha_F = 0$. Consequently, the step α_k will be zero and $x_{k+1} = x_k$, resulting in no change in the objective. However, the working set does change with $\mathcal{W}_{k+1}$ differing from $\mathcal{W}_k$ by the addition of one blocking active constraint.

Constraint deletion and addition. The following results show the effects of deleting and adding constraints on stationarity and the reduced Hessian. The first shows the effect of the deletion of a constraint from $\mathcal{W}_k$ at a subspace minimizer. The second considers the effects of adding a blocking constraint to the working set.

Result 2.2.1 (Constraint deletion). *Let x_{k-1} be a subspace minimizer with working set $\mathcal{W}_{k-1}$. Define $x_k = x_{k-1}$ and $\mathcal{W}_k = \mathcal{W}_{k-1} - \{\nu_s\}$. For simplicity, assume that the working-set matrix has the form*

$$A_{k-1} = \begin{pmatrix} A_k \\ d_{\nu_s}^T \end{pmatrix}.$$

Then x_k and $\mathcal{W}_k$ satisfy the following:

(a) *$g_k = A_k^T y_k + \sigma d_{\nu_s}$ for some vector y_k and $\sigma < 0$; and*

(b) *the reduced Hessian $Z_k^T H Z_k$ has at most one nonpositive eigenvalue.*

Proof. Part (a) holds from (2.3).

For part (b), let the columns of Z_{k-1} form a basis for the null space of A_{k-1}. Then $A_k Z_{k-1} = 0$ and Z_{k-1} can be extended to form a basis for the null space of A_k, with $Z_k = \begin{pmatrix} Z_{k-1} & z \end{pmatrix}$. Then,

$$Z_k^T H Z_k = \begin{pmatrix} Z_{k-1}^T H Z_{k-1} & Z_{k-1}^T H z \\ z^T H Z_{k-1} & z^T H z \end{pmatrix}.$$

Let $\{\lambda_j\}$ denote the eigenvalues of $Z_k^T H Z_k$ with $\lambda_j \le \lambda_{j-1}$. Similarly, let $\{\lambda_j^-\}$ denote the eigenvalues of $Z_{k-1}^T H Z_{k-1}$ with $\lambda_j^- \le \lambda_{j-1}^-$. The interlacing eigenvalue property (Result 1.3.5) implies that

$$\lambda_{n-(m+m_k+1)} \ge \lambda_{n-(m+m_k+1)}^- \ge \lambda_{n-(m+m_k)}.$$

Since $Z_{k-1}^T H Z_{k-1}$ is positive definite, $\lambda_{n-(m+m_k+1)}^- > 0$ and $Z_k^T H Z_k$ has at most one nonpositive eigenvalue. $\square$

Result 2.2.2 (Constraint addition). *Suppose that $d_r^T x \ge f_r$ is a blocking constraint at $x_{k+1} = x_k + \alpha_k p_k$. Let $\mathcal{W}_{k+1} = \mathcal{W}_k + \{r\}$ and assume that ν_s is the index of the most recently deleted constraint. Define the matrix Z_k such that its columns form a basis for null space for A_k. Then x_{k+1} and $\mathcal{W}_{k+1}$ satisfy the following:*

(a) *$g_{k+1} = A_{k+1}^T y_{k+1} + \sigma d_{\nu_s}$ for some $\sigma < 0$;*

(b) *the reduced Hessian $Z_{k+1}^T H Z_{k+1}$ has at most one nonpositive eigenvalue; and*

(c) *the set $\mathcal{W}_{k+1} + \{\nu_s\}$ is a second-order-consistent working set.*

Proof. If p_k is defined by (2.6), then (2.7) implies $g_{k+1} = A_k^T(y_k + \alpha q_k) + (1-\alpha)[y_k]_{m+s} d_{\nu_s}$ with $(1-\alpha)[y_k]_{m+s} < 0$ since $\alpha < 1$. Otherwise, p_k is defined by (2.8) and (2.9) holds. The desired result follows by induction.

For part (b), let the columns of Z_k form a null space for A_k and denote the KKT matrices associated with $\mathcal{W}_k$ and $\mathcal{W}_{k+1}$ as

$$K = \begin{pmatrix} H & A_k^T \\ A_k & 0 \end{pmatrix} \quad \text{and} \quad K^+ = \begin{pmatrix} H & A_k^T & d_r \\ A_k & 0 & 0 \\ d_r^T & 0 & 0 \end{pmatrix}.$$

Assume that K has eigenvalues $\{\lambda_j\}$ with $\lambda_j \geq \lambda_{j-1}$. Similarly, K^+ has eigenvalues $\{\lambda_j^+\}$ with $\lambda_j^+ \geq \lambda_{j-1}^+$. The eigenvalue interlacing property (Result 1.3.5) implies

$$\lambda_{n-1}^+ \geq \lambda_{n-1} \geq \lambda_n^+ \geq \lambda_n \geq \lambda_{n+1}^+. \tag{2.14}$$

Since A_k has full row rank, Corollary 1.3.1 implies that $\mathrm{In}(K) = \mathrm{In}(Z_k^T H Z_k) + (m + m_k, m + m_k, 0)$. If $Z_k^T H Z_k$ is positive definite, then $\mathrm{In}(K) = (n, m + m_k, 0)$ and it must hold that $\lambda_n > 0$. The equation (2.14) implies that $\lambda_n^+ > 0$, so that K^+ has at least n positive eigenvalues and at most $m + m_k + 1$ nonpositive eigenvalues. Thus, since the inertia of K^+ satisfies the relation $\mathrm{In}(K^+) = \mathrm{In}(Z_{k+1}^T H Z_{k+1}) + (m + m_k + 1, m + m_k + 1, 0)$, then $Z_{k+1}^T H Z_{k+1}$ is positive definite.

If $Z_k^T H Z_k$ has one nonpositive eigenvalue, then $\lambda_{n-1} > 0$ and K^+ has at least $n - 1$ positive eigenvalues and at most $m + m_k + 2$ nonpositive eigenvalues. Thus, $Z_{k+1}^T H Z_{k+1}$ has at most one nonpositive eigenvalue. $\square$

Thus far, we have only established that a subspace minimizer is reached when the reduced Hessian is positive definite and an unconstrained step is taken. It remains to show that if a subspace stationary point is reached by taking a blocking step and adding a constraint to the working set, then that point is also a subspace minimizer.

Result 2.2.3 (Subspace minimizer with blocking constraint)**.** *Let $\mathcal{W}_k$ be a working set such that $Z_k^T H Z_k$ is nonsingular. Assume that the constraint with index ν_s is deleted from the working set and p_k is defined by (2.6). Suppose that $d_r^T x \geq f_r$ is a blocking constraint at $x_k + \alpha_k p_k$, where $\alpha_k < 1$. Let $\mathcal{W}_{k+1} = \mathcal{W}_k + \{r\}$.*

(a) *The point $x_k + \alpha_k p_k$ is stationary with respect to $\mathcal{W}_{k+1}$ if and only if d_r is linearly dependent on the rows of A_k and d_{ν_s}.*

(b) *If $x_k + \alpha_k p_k$ is a stationary point with respect to $\mathcal{W}_{k+1}$, then $x_k + \alpha_k p_k$ is a subspace minimizer with respect to $\mathcal{W}_{k+1}$.*

Proof. Suppose that $x_k + \alpha_k p_k$ is a stationary point with respect to $\mathcal{W}_{k+1}$. Then there exist a vector v and nonzero scalar σ such that $g(x_k + \alpha_k p_k) = A_k^T v + \sigma d_r$. However, (2.7) implies that $g(x_k + \alpha_k p_k) = A_k^T(y_k + \alpha_k q_k) + (1 - \alpha_k)[y_k]_{m+s} d_{\nu_s}$. Eliminating $g(x_k + \alpha_k p_k)$ yields

$$A_k^T(y_k + \alpha_k q_k - v) + (1 - \alpha_k)[y_k]_{m+s} d_{\nu_s} = \sigma d_r.$$

Since $\alpha_k < 1$, d_r is linearly dependent on the rows of A_k and d_{ν_s}.

Now suppose that d_r is linearly dependent on the rows of A_k and d_{ν_s}, with $d_r = \sigma d_{\nu_s} + A_k^T v$ and $\sigma \neq 0$. Then by (2.7),

$$\begin{aligned} g(x_k + \alpha_k p_k) &= A_k^T(y_k + \alpha_k q_k) + (1 - \alpha_k)\frac{1}{\sigma}(d_r - A_k^T v) \\ &= \frac{1}{\sigma}(1 - \alpha_k)[y_k]_{m+s} d_r + A_k^T(y_k + \alpha_k q_k - \frac{1}{\sigma}(1 - \alpha_k)[y_k]_{m+s} v). \end{aligned}$$

$$
\begin{array}{lcccccccccc}
 & x_0 & \longrightarrow & x_1 & \longrightarrow & x_2 & \longrightarrow & \cdots & x_k & \longrightarrow & x_{k+1} \\[2ex]
\text{(A)} & \mathcal{W}_0 & \xrightarrow{\textit{delete } \nu_s} & \mathcal{W}_1 & \xrightarrow{\textit{move, add}} & \mathcal{W}_2 & \xrightarrow{\textit{move, add}} & \cdots & \mathcal{W}_k & \xrightarrow{\textit{move}} & \mathcal{W}_{k+1} \\[2ex]
\text{(B)} & \mathcal{W}_0 & \xrightarrow{\textit{delete } \nu_s} & \mathcal{W}_1 & \xrightarrow{\textit{move, add}} & \mathcal{W}_2 & \xrightarrow{\textit{move, add}} & \cdots & \mathcal{W}_k & \xrightarrow{\textit{move, add}} & \mathcal{W}_{k+1}
\end{array}
$$

Figure 2.1: This figure depicts the two types of sequence of consecutive iterates in the binding-direction method. Each sequence starts and ends with subspace minimizers x_0 and x_{k+1}, with intermediate iterates that are not subspace minimizers. The sequences differ in how the final point is reached. In (A), an unconstrained step is taken ($\alpha = 1$). In (B), a blocking step ($\alpha_F < \alpha_*$) is taken, and a blocking constraint is added to the working set that makes the reduced Hessian positive definite and hence, makes x_{k+1} a subspace minimizer.

Again, $\alpha_k < 1$, $\sigma \neq 0$ and $y_{m+s} < 0$, so that $x_k + \alpha_k p_k$ is a stationary point with respect to $\mathcal{W}_{k+1}$.

For part (b), if z is in the null space of A_{k+1}, then $A_k z = 0$ and $d_r^T z = 0$. However, by part (a), d_r must be linearly dependent on the rows of A_k and d_{ν_s}. Therefore,

$$
0 = d_r^T z = \begin{pmatrix} v^T & \sigma \end{pmatrix} \begin{pmatrix} A_k \\ d_{\nu_s}^T \end{pmatrix} z = \sigma d_{\nu_s}^T z.
$$

Since $\sigma \neq 0$, $d_{\nu_s}^T z = 0$ and z lies in the null space of $\begin{pmatrix} A_k \\ d_{\nu_s}^T \end{pmatrix}$. By part (c) of Result 2.2.2, the reduced Hessian associated with this matrix is positive definite. Therefore, $Z_{k+1}^T H Z_{k+1}$ is positive definite and $x_k + \alpha_k p_k$ is a subspace minimizer with respect to $\mathcal{W}_{k+1}$. $\square$

Algorithm Summary. Given an arbitrary feasible point x_0, and an initial second-order-consistent working set $\mathcal{W}_0$, the procedure defined generates a sequence of points $\{x_k\}$ and associated working sets $\mathcal{W}_k$ such that $x_{k+1} = x_k + \alpha_k p_k$, where p_k is computed from either (2.6) or (2.8). Because a constraint cannot be deleted until a subspace minimizer is found, the algorithm starts by adding constraints to the working set until either an unconstrained step is taken ($\alpha_k = 1$) or sufficiently many constraints are added to define a subspace minimizer (e.g., at a vertex, which is trivially a subspace minimizer). Once the first subspace minimizer is found, the iterates occur in groups of consecutive iterates where each group starts with a constraint deletion and ends with a step to a subspace minimizer. Figure 2.1 illustrates the two ways that the algorithm arrives at a subspace minimizer.

At every iteration, either x or the working set changes, giving a sequence of distinct pairs $\{x_k, \mathcal{W}_k\}$, where $x_{k+1} \neq x_k$ or $\mathcal{W}_{k+1} \neq \mathcal{W}_k$. With a suitable nondegeneracy assumption, the algorithm terminates in a finite number of iterations. Since the number of constraints is finite, the sequence $\{x_k\}$ must contain a subsequence $\{x_{ik}\}$ of subspace minimizers with respect to their working sets $\{\mathcal{W}_{ik}\}$. If the Lagrange multipliers are nonnegative at any of these points, the algorithm terminates with the desired solution. Otherwise, at least one multiplier must be strictly negative, and hence the nondegeneracy assumption implies that $\alpha_F > 0$ at x_{ik}. Thus, $\varphi(x_{ik}) > \varphi(x_{ik} + \alpha_{ik} p_{ik})$, since at each iteration, the direction is defined as a descent direction with $g^T p < 0$. The subsequence $\{x_{ik}\}$ must be finite because the number of subspace minimizers is finite and the strict decrease in $\varphi(x)$ guarantees that no element of $\{x_{ik}\}$ is repeated. The finiteness of the subsequence implies that the number of intermediate iterates must also be finite. This follows because a constraint is added to the working set (possibly with a zero step) for every intermediate iteration. Eventually, either a nonzero step will be taken, giving a strict decrease in φ, or enough constraints will be added to define a vertex (a trivial subspace minimizer).

Algorithm 2.1: Binding-direction method for general QP

Find x such that $Ax = b$, $Dx \geq f$;
Choose $\mathcal{W} \subseteq \mathcal{A}(x)$ such that the working-set matrix has full row rank;
$\big[x, \mathcal{W}_0\big] = \texttt{subspaceMin}(x, H, A, D, \mathcal{W})$;
$k = 0$; $\quad g = c + Hx$;
repeat
    while $k > 0$ and $g \neq A^T\pi + D_k^T z$ **do**
        $\big[p, q\big] = \texttt{descent\_direction}(D_k, A, H)$;
        $\alpha_F = \texttt{maxStep}(x, p, D, f)$;
        if $p^T Hp > 0$ **then** $\alpha_* = 1$ **else** $\alpha_* = +\infty$;
        $\alpha = \min\{\alpha_*, \alpha_F\}$;
        if $\alpha = +\infty$ **then stop**; [the solution is unbounded]
        if $\alpha_F < \alpha_*$ **then** [add a blocking constraint]
            Choose a blocking constraint index t; $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k + \{t\}$;
        end;
        $x \leftarrow x + \alpha p$; $\quad g \leftarrow g + \alpha Hp$;
        $k \leftarrow k + 1$;
    end do;
    Solve $g = \begin{pmatrix} A^T & D_k^T \end{pmatrix} \begin{pmatrix} \pi \\ z \end{pmatrix}$; $\quad s = \mathrm{argmin}_i\{z_i\}$;
    if $z_s < 0$ **then** [delete a constraint]
        $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k - \{\nu_s\}$; $\quad k \leftarrow k + 1$;
    end;
until $z_s \geq 0$;

The binding-direction algorithm is summarized in Algorithm 2.1. The `subspaceMin` function computes an initial point and basis (see Section 5.2). The function `maxStep` simply computes the maximum feasible step, while the direction p is computed by an appropriate "black box" function `descent_direction`.

2.2.2 Nonbinding-direction method

A feature of the binding-direction method is that the reduced Hessian may have one nonpositive eigenvalue, which precludes the use of the Cholesky factorization $Z_k^T H Z_k = R_k^T R_k$. In this section, the *nonbinding-direction method* is introduced as an active-set method that keeps the reduced Hessian positive definite (and hence keeps the KKT matrices nonsingular) allowing for the efficient calculation of search directions.

As in the binding-direction method, the nonbinding-direction method starts at a standard subspace minimizer x, i.e., $g(x) = A_w^T y = A^T\pi + D_w^T z_w$ and $\mathrm{In}(K) = (n, m + m_w, 0)$. Let ν_s be an index in the working set such that $[z_w]_s < 0$. To proceed, a descent direction is defined that is feasible for the equality constraints and the constraints in the working set. Analogous to (2.2), p is defined so that

$$g(x)^T p < 0 \ \text{ and } \ A_w p = e_{m+s}.$$

Unlike the binding-direction method, the direction p is computed without removing ν_s from the working set. As any nonzero step along p must increase the residual of the ν_s-th constraint (thereby making it inactive or nonbinding), the working set is no longer a subset of the active set. The direction is defined as the solution of the equality-constrained subproblem

$$\underset{p}{\text{minimize}} \ \ \varphi(x+p) \quad \text{subject to} \quad A_w p = e_{m+s}. \tag{2.15}$$

The optimality conditions for this subproblem imply the existence of a vector q such that $g(x + p) = A_w^T(y + q)$; i.e., q is the step to the multipliers associated with the optimal solution $x + p$. This condition, along with the feasibility condition, implies that p and q satisfy the equations

$$\begin{pmatrix} H & A_w^T \\ A_w & 0 \end{pmatrix} \begin{pmatrix} p \\ -q \end{pmatrix} = \begin{pmatrix} -(g(x) - A_w^T y) \\ e_{m+s} \end{pmatrix}.$$

Important properties of the primal and dual vectors are summarized in the next result.

Result 2.2.4 (Properties of a nonbinding search direction)**.** *Let x be a subspace minimizer such that $g = A_w^T y = A^T\pi + D_w^T z_w$, with $[z_w]_s < 0$. Then the vectors p and q satisfying the equations*

$$\begin{pmatrix} H & A_w^T \\ A_w & 0 \end{pmatrix} \begin{pmatrix} p \\ -q \end{pmatrix} = \begin{pmatrix} -(g(x) - A_w^T y) \\ e_{m+s} \end{pmatrix} = \begin{pmatrix} 0 \\ e_{m+s} \end{pmatrix} \tag{2.16}$$

constitute the unique primal and dual solutions of the equality constrained problem defined by minimizing $\varphi(x+p)$ subject to $A_w p = e_{m+s}$. Moreover, p and q satisfy the identities

$$g^T p = y_{m+s} = [z_w]_s \quad \textit{and} \quad p^T H p = q_{m+s} = [q_w]_s, \tag{2.17}$$

where q_w denotes the vector of last m_w components of q.

Proof. The assumption that x is a subspace minimizer implies that the subproblem (2.15) has a unique bounded minimizer. The optimality of p and q follows from the equations in (2.16), which represent the feasibility and optimality conditions for the minimization of $\varphi(x+p)$ on the set $\{p : A_w\, p = e_{m+s}\}$. The equation $g = A_w^T y$ and the definition of p from (2.16) give

$$g^T p = p^T(A_w^T y) = y^T A_w\, p = y^T e_{m+s} = y_{m+s} = [z_w\,]_s.$$

Similarly, $p^T H p = p^T(A_w^T q) = e_{m+s}^T q = q_{m+s} = [q_w\,]_s$. □

Once p and q are known, a nonnegative step α is computed so that $x+\alpha p$ is feasible and $\varphi(x+\alpha p) \le \varphi(x)$. If $p^T H p > 0$, the step that minimizes $\varphi(x+\alpha p)$ as a function of α is given by $\alpha_* = -g^T p/p^T H p$. The identities (2.17) give

$$\alpha_* = -g^T p/p^T H p = -[z_w\,]_s/[q_w\,]_s. \tag{2.18}$$

Since $[z_w\,]_s < 0$, if $[q_w\,]_s = p^T H p > 0$, the optimal step α_* is positive. Otherwise $[q_w\,]_s = p^T H p \le 0$ and φ has no bounded minimizer along p and $\alpha_* = +\infty$.

The maximum feasible step is computed as in (2.13) to limit α in case the optimal step is unbounded or infeasible. The step α is then $\min\{\alpha_*, \alpha_F\}$. If $\alpha = +\infty$, the QP has no bounded solution and the algorithm terminates. In the discussion below, we assume that α is a bounded step.

The primal and dual directions p and q defined by (2.16) have the property that $x+\alpha p$ remains a subspace minimizer with respect to A_w *for any step* α. This follows from the equations (2.16), which imply that

$$g(x+\alpha p) = g(x) + \alpha H p = A_w^T y + \alpha A_w^T q = A_w^T(y+\alpha q), \tag{2.19}$$

so that the gradient at $x+\alpha p$ is a linear combination of the columns of A_w^T. The step $x+\alpha p$ does not change the KKT matrix K associated with the subspace minimizer x, which implies that $x+\alpha p$ is also a subspace minimizer with respect to A_w. This means that $x+\alpha p$ may be interpreted as the solution of a problem in which the working-set constraint $d_{\nu_s}^T x \ge f_{\nu_s}$ is *shifted* to pass through $x+\alpha p$. The component $[y+\alpha q]_{m+s} = [z_w + \alpha q_w\,]_s$ is the Lagrange multiplier associated with the shifted version of $d_{\nu_s}^T x \ge f_{\nu_s}$. This property is known as the *parallel subspace property* of quadratic programming. It shows that if x is stationary with respect to a nonbinding constraint, then it remains so for all subsequent iterates for which that constraint remains in the working set.

Once α has been defined, the new iterate is $\bar{x} = x + \alpha p$. The composition of the new working set and multipliers depends on the definition of α.

Case 1: $\alpha = \alpha_*$ In this case, $\alpha = \alpha_* = -[z_w]_s/[q_w]_s$ minimizes $\varphi(x+\alpha p)$ with respect to α, giving the s-th element of $z_w + \alpha q_w$ as

$$[z_w + \alpha q_w]_s = [z_w]_s + \alpha_*[q_w]_s = 0,$$

which implies that the Lagrange multiplier associated with the shifted constraint is *zero* at $\bar{x}$. The nature of the stationarity may be determined using the next result.

Result 2.2.5 (Constraint deletion)**.** *Let x be a subspace minimizer with respect to $\mathcal{W}$. Assume that $[z_w]_s < 0$. Let $\bar{x}$ denote the point $x+\alpha p$, where p is defined by* (2.16) *and $\alpha = \alpha_*$ is bounded. Then $\bar{x}$ is a subspace minimizer with respect to $\bar{\mathcal{W}} = \mathcal{W} - \{\nu_s\}$.*

Proof. Let K and $\bar{K}$ denote the matrices

$$K = \begin{pmatrix} H & A_w^T \\ A_w & \end{pmatrix} \quad \text{and} \quad \bar{K} = \begin{pmatrix} H & \bar{A}_w^T \\ \bar{A}_w & \end{pmatrix},$$

where A_w and $\bar{A}_w$ are the working-set matrices associated with $\mathcal{W}$ and $\bar{\mathcal{W}}$. It suffices to show that $\bar{K}$ has the correct inertia, i.e., $\operatorname{In}(\bar{K}) = (n, m + m_w - 1, 0)$.

Consider the matrix M such that

$$M \triangleq \begin{pmatrix} K & e_{m+n+s} \\ e_{m+n+s}^T & \end{pmatrix}.$$

By assumption, x is a subspace minimizer with $\operatorname{In}(K) = (n, m + m_w, 0)$. In particular, K is nonsingular and the Schur complement of K in M exists with

$$M/K = -e_{n+m+s}^T K^{-1} e_{n+m+s} = -e_{n+m+s}^T \begin{pmatrix} p \\ -q \end{pmatrix} = [q_w]_s.$$

It follows that

$$\operatorname{In}(M) = \operatorname{In}(M/K) + \operatorname{In}(K) = \operatorname{In}([q_w]_s) + (n, m + m_w, 0). \tag{2.20}$$

Now consider a symmetrically permuted version of M:

$$\widetilde{M} = \begin{pmatrix} 0 & 1 & & \\ 1 & 0 & d_{\nu_s}^T & \\ & d_{\nu_s} & H & \bar{A}_w^T \\ & & \bar{A}_w & \end{pmatrix}. \tag{2.21}$$

Inertia is unchanged by symmetric permutations, so $\operatorname{In}(M) = \operatorname{In}(\widetilde{M})$. The 2×2 block in the upper-left corner of $\widetilde{M}$, denoted by E, has eigenvalues ± 1, so that $\operatorname{In}(E) = (1, 1, 0)$ with $E^{-1} = E$. The Schur complement of E in $\widetilde{M}$ is

$$\widetilde{M}/E = \bar{K} - \begin{pmatrix} 0 & d_{\nu_s} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ d_{\nu_s}^T & 0 \end{pmatrix} = \bar{K}, \tag{2.22}$$

which implies that $\mathrm{In}(\widetilde{M}) = \mathrm{In}(\widetilde{M}/E) + \mathrm{In}(E) = \mathrm{In}(\bar{K}) + (1,1,0)$. Combining this with (2.20) yields

$$\begin{aligned} \mathrm{In}(\bar{K}) &= \mathrm{In}([q_w]_s) + (n, m+m_w, 0) - (1,1,0) \\ &= \mathrm{In}([q_w]_s) + (n-1, m+m_w-1, 0). \end{aligned}$$

Since $\alpha = \alpha_*$, $[q_w]_s$ must be positive. It follows that

$$\mathrm{In}(\bar{K}) = (1,0,0) + (n-1, m+m_w-1, 0) = (n, m+m_w-1, 0)$$

and the subspace stationary point $\bar{x}$ is a (standard) subspace minimizer with respect to the new working set $\bar{\mathcal{W}} = \mathcal{W} - \{\nu_s\}$. □

Case 2: $\alpha = \alpha_F$ In this case, α is the step to the blocking constraint $d_r^T x \geq f_r$, which is eligible to be added to the working set at $x + \alpha p$. However, the definition of the new working set depends on whether or not the blocking constraint is dependent on the constraints already in $\mathcal{W}$. If d_r is linearly independent of the columns of A_w^T, then the index r is added to the working set. Otherwise, we show in Result 2.2.7 below that a suitable working set is defined by exchanging rows d_{ν_s} and d_r in A_w. The following result provides a computable test for the independence of d_r and the columns of A_w^T.

Result 2.2.6 (Test for constraint dependency)**.** *Assume that x is a subspace minimizer with respect to A_w. Assume that $d_r^T x \geq f_r$ is a blocking constraint at $\bar{x} = x + \alpha p$, where p satisfies* (2.16)*. Let vectors u and v be the solutions of the system*

$$\begin{pmatrix} H & A_w^T \\ A_w & \end{pmatrix} \begin{pmatrix} u \\ -v \end{pmatrix} = \begin{pmatrix} d_r \\ 0 \end{pmatrix}, \tag{2.23}$$

then

(a) *the vector d_r and the columns of A_w^T are linearly independent if and only if $u \neq 0$;*

(b) $v_{m+s} = -d_r^T p > 0$*, and if $u \neq 0$, then $u^T d_r > 0$.*

Proof. For part (a), equations (2.23) give $Hu - A_w^T v = d_r$ and $A_w u = 0$. If $u = 0$ then $-A_w^T v = d_r$, and d_r must be dependent on the columns of A_w^T. Conversely, if $-A_w^T v = d_r$, then the definition of u gives $u^T d_r = -u^T A_w^T v = 0$, which implies that $u^T H u = u^T(Hu - A_w^T v) = u^T d_r = 0$. By assumption, x is a subspace minimizer with respect to A_w, which is equivalent to the assumption that H is positive definite for all u such that $A_w u = 0$. Hence $u^T H u = 0$ can hold only if u is zero.

For part (b), we use equations (2.16) and (2.23) to show that

$$v_{m+s} = e_{m+s}^T v = p^T A_w^T v = p^T(Hu - d_r) = q^T A_w u - p^T d_r = -d_r^T p > 0,$$

where the final inequality follows from the fact that $d_r^T p$ must be negative if $d_r^T x \geq f_r$ is a blocking constraint.

Equations (2.23) imply $Hu - A_w^T v = d_r$ and $A_w u = 0$. Multiplying the first equation by u^T and applying the second equation gives $u^T H u = u^T d_r$. As x is a subspace minimizer and u is nonzero with $u \in \text{null}(A_w)$, it must hold that $u^T H u = u^T d_r > 0$, as required. $\square$

The next result provides expressions for the updated multipliers.

Result 2.2.7 (Multiplier updates)**.** *Assume that x is a subspace minimizer with respect to A_w. Assume that $d_r^T x \geq f_r$ is a blocking constraint at the next iterate $\bar{x} = x + \alpha p$, where the direction p satisfies* (2.16)*. Let u and v satisfy* (2.23)*.*

(a) *If d_r and the columns of A_w^T are linearly independent, then the vector $\bar{y}$ formed by appending a zero to the vector $y + \alpha q$ satisfies $g(\bar{x}) = \bar{A}_w^T \bar{y}$, where $\bar{A}_w$ denotes the matrix A_w with row d_r^T added in the last position.*

(b) *If d_r and the columns of A_w^T are linearly dependent, then the vector $\bar{y}$ such that*

$$\bar{y} = y + \alpha q + \sigma v, \quad \textit{with} \quad \sigma = -[y + \alpha q]_{m+s}/v_{m+s}, \tag{2.24}$$

satisfies $g(\bar{x}) = A_w^T \bar{y} + \sigma d_r$ with $\bar{y}_{m+s} = 0$ and $\sigma > 0$.

Proof. For part (a), the identity (2.19) implies that $g(x + \alpha p) = g(\bar{x}) = A_w^T (y + \alpha q)$. As d_r and the columns of A_w^T are linearly independent, we may add the index r to $\mathcal{W}$ and define the new working-set matrix $\bar{A}_w^T = \begin{pmatrix} A_w^T & d_r \end{pmatrix}$. This allows us to write $g(\bar{x}) = \bar{A}_w^T \bar{y}$, with $\bar{y}$ given by $y + \alpha q$ with an appended zero component.

Now assume that A_w^T and d_r are linearly dependent. From Result 2.2.6 it must hold that $u = 0$ and there exists a unique v such that $d_r = -A_w^T v$. For any value of σ, it holds that

$$g(\bar{x}) = A_w^T (y + \alpha q) = A_w^T (y + \alpha q + \sigma v) + \sigma d_r.$$

If we choose $\sigma = -[y + \alpha q]_{m+s}/v_{m+s}$ and define the vector $\bar{y} = y + \alpha q + \sigma v$, then

$$g(\bar{x}) = A_w^T \bar{y} + \sigma d_r, \quad \text{with} \quad \bar{y}_{m+s} = [y + \alpha q + \sigma v]_{m+s} = 0.$$

It follows that $g(\bar{x})$ is a linear combination of d_r and every column of A_w^T except d_{ν_s}.

In order to show that $\sigma = -[y + \alpha q]_{m+s}/v_{m+s}$ is positive, consider the linear function $\eta(\alpha) = [y + \alpha q]_{m+s}$, which satisfies $\eta(0) = y_{m+s} < 0$. If $q_{m+s} = p^T H p > 0$, then $\alpha_* < \infty$ and $\eta(\alpha)$ is an increasing linear function of positive α with $\eta(\alpha_*) = 0$. This implies that $\eta(\alpha) < 0$ for any $\alpha < \alpha_*$ and $\eta(\alpha_k) < 0$. If $q_{m+s} \leq 0$, then $\eta(\alpha)$ is a non-increasing linear function of α so that $\eta(\alpha) < 0$ for any positive α. Thus, $[y + \alpha q]_{m+s} < 0$ for any $\alpha < \alpha_*$, and $\sigma = -[y + \alpha q]_{m+s}/v_{m+s} > 0$ from part (b) of Result 2.2.6. $\square$

Result 2.2.8. *Let x be a subspace minimizer with respect to the working set $\mathcal{W}$. Assume that $d_r^T x \geq f_r$ is a blocking constraint at $\bar{x} = x + \alpha p$, where p is defined by* (2.16).

(a) *If d_r is linearly independent of the columns of A_w^T, then $\bar{x}$ is a subspace minimizer with respect to the working set $\bar{\mathcal{W}} = \mathcal{W} + \{r\}$.*

(b) *If d_r is dependent on the columns of A_w^T, then $\bar{x}$ is a subspace minimizer with respect to the working set $\bar{\mathcal{W}} = \mathcal{W} + \{r\} - \{\nu_s\}$.*

Proof. Parts (a) and (b) of Result 2.2.7 imply that $\bar{x}$ is a subspace stationary point with respect to $\bar{\mathcal{W}}$. It remains to show that in each case, the KKT matrix for the new working set has correct inertia.

For part (a), it suffices to show that the KKT matrix for the new working set $\bar{\mathcal{W}} = \mathcal{W} + \{r\}$ has inertia $(n, m + m_w + 1, 0)$. Assume that d_r and the columns of A_w^T are linearly independent, so that the vector u of (2.23) is nonzero. Let K and $\bar{K}$ denote the KKT matrices associated with the working sets $\mathcal{W}$ and $\bar{\mathcal{W}}$, i.e.,

$$K = \begin{pmatrix} H & A_w^T \\ A_w & \end{pmatrix} \quad \text{and} \quad \bar{K} = \begin{pmatrix} H & \bar{A}_w^T \\ \bar{A}_w & \end{pmatrix},$$

where $\bar{A}_w$ is the matrix A_w with the row d_r^T added in the last position.

By assumption, x is a subspace minimizer and $\mathrm{In}(K) = (n, m + m_w, 0)$. It follows that K is nonsingular and the Schur complement of K in $\bar{K}$ exists with

$$\bar{K}/K = -\begin{pmatrix} d_r^T & 0 \end{pmatrix} K^{-1} \begin{pmatrix} d_r \\ 0 \end{pmatrix} = -\begin{pmatrix} d_r^T & 0 \end{pmatrix} \begin{pmatrix} u \\ -v \end{pmatrix} = -d_r^T u < 0,$$

where the last inequality follows from part (b) of Result 2.2.6. Then,

$$\begin{aligned} \mathrm{In}(\bar{K}) = \mathrm{In}(\bar{K}/K) + \mathrm{In}(K) &= \mathrm{In}(-u^T d_r) + (n, m + m_w, 0) \\ &= (0, 1, 0) + (n, m + m_w, 0) = (n, m + m_w + 1, 0). \end{aligned}$$

For part (b), assume that d_r and the columns of A_w^T are linearly dependent and that $\bar{\mathcal{W}} = \mathcal{W} + \{r\} - \{\nu_s\}$. By Result 2.2.7 and equation (2.23), it must hold that $u = 0$ and $-A_w^T v = d_r$. Let A_w and $\bar{A}_w$ be the working-set matrices associated with $\mathcal{W}$ and $\bar{\mathcal{W}}$. The change in the working set replaces row s of D_w by d_r^T, so that

$$\begin{aligned} \bar{A}_w = A_w + e_{m+s}(d_r^T - d_{\nu_s}^T) &= A_w + e_{m+s}(-v^T A_w - e_{m+s}^T A_w) \\ &= \big(I - e_{m+s}(v + e_{m+s})^T\big) A_w \\ &= M A_w, \end{aligned}$$

where $M = I - e_{m+s}(v + e_{m+s})^T$. The matrix M has $m + m_w - 1$ unit eigenvalues and one eigenvalue equal to v_{m+s}. From part (b) of Result 2.2.6, it holds that $v_{m+s} > 0$ and hence M is

$$\begin{array}{ccccccccc}
 & x_0 & \longrightarrow & \cdots & x_{k-1} & \longrightarrow & x_k & \longrightarrow & x_{k+1} \\
 & & & & & & & & \\
\text{(A)} & \mathcal{W}_0 & \xrightarrow{\textit{move, add}} & \cdots & \mathcal{W}_{k-1} & \xrightarrow{\textit{move, add}} & \mathcal{W}_k & \xrightarrow{\textit{move and delete } \nu_s} & \mathcal{W}_{k+1} \\
 & & & & & & & & \\
\text{(B)} & \mathcal{W}_0 & \xrightarrow{\textit{move, add}} & \cdots & \mathcal{W}_{k-1} & \xrightarrow{\textit{move, add}} & \mathcal{W}_k & \xrightarrow{\textit{move and swap}} & \mathcal{W}_{k+1}
\end{array}$$

Figure 2.2: Each sequence starts and ends with a standard subspace minimizer x_0 and x_{k+1}, with intermediate iterates that are nonstandard subspace minimizers. In (A), x_{k+1} is reached by taking an optimal step and the ν_s-th constraint is removed from the working set. In (B), a linearly dependent blocking constraint is swapped with the ν_s-th constraint making x_{k+1} a standard subspace minimizer.

nonsingular. The new KKT matrix for $\bar{\mathcal{W}}$ can be written as

$$\begin{pmatrix} H & \bar{A}_w^T \\ \bar{A}_w & \end{pmatrix} = \begin{pmatrix} I & \\ & M \end{pmatrix} \begin{pmatrix} H & A_w^T \\ A_w & \end{pmatrix} \begin{pmatrix} I & \\ & M^T \end{pmatrix}.$$

By Sylvester's Law of Inertia, the old and new KKT matrices have the same inertia, which implies that $\bar{x}$ is a subspace minimizer with respect to $\bar{\mathcal{W}}$. $\square$

The first part of this result shows that $\bar{x}$ is a subspace minimizer both before and after an independent constraint is added to the working set. This is crucial because it means that the directions p and q for the next iteration satisfy the KKT equations (2.16) with $\bar{A}_w$ in place of A_w. The second part shows that the working-set constraints can be linearly dependent only at a *standard* subspace minimizer associated with a working set that does not include constraint ν_s. This implies that it is appropriate to remove ν_s from the working set. The constraint $d_{\nu_s}^T x \geq f_{\nu_s}$ plays a significant (and explicit) role in the definition of the search direction and is called the *nonbinding working-set constraint.* The method generates sets of consecutive iterates that begin and end with a standard subspace minimizer. The nonbinding working-set constraint $d_{\nu_s}^T x \geq f_{\nu_s}$ identified at the first point of the sequence is deleted from the working set at the last point (either by deletion or replacement).

The proposed method is the basis for Algorithm 2.2 given below. Each iteration requires the solution of two KKT systems:

$$\text{Full System 1} \qquad \begin{pmatrix} H & A_w^T \\ A_w & 0 \end{pmatrix} \begin{pmatrix} p \\ -q \end{pmatrix} = \begin{pmatrix} 0 \\ e_{m+s} \end{pmatrix} \tag{2.25a}$$

$$\text{Full System 2} \qquad \begin{pmatrix} H & A_w^T \\ A_w & 0 \end{pmatrix} \begin{pmatrix} u \\ -v \end{pmatrix} = \begin{pmatrix} d_r \\ 0 \end{pmatrix}. \tag{2.25b}$$

However, for those iterations for which the number of constraints in the working set increases, it is possible to *update* the vectors p and q, making it unnecessary to solve (2.25a).

Algorithm 2.2: Nonbinding-direction method for general QP

Find x such that $Ax = b$, $Dx \geq f$; $\quad k = 0$;

Choose $\mathcal{W}$, any full-rank subset of $\mathcal{A}(x)$; $\quad$ Choose π and z_w;

$\big[x, \pi, z_w, \mathcal{W}\big] = \texttt{subspaceMin}(x, \pi, z_w, \mathcal{W})$; $\quad m_w = |\mathcal{W}|$;

$g = c + Hx$; $\quad s = \operatorname{argmin}_i\, [z_w]_i$;

while $[z_w]_s < 0$ **do**

Solve $\begin{pmatrix} H & A^T & D_w^T \\ A & 0 & 0 \\ D_w & 0 & 0 \end{pmatrix} \begin{pmatrix} p \\ -q_\pi \\ -q_w \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ e_s \end{pmatrix}$;

$\alpha_F = \texttt{maxStep}(x, p, D, f)$;

if $[q_w]_s > 0$ **then** $\alpha_* = -[z_w]_s/[q_w]_s$ **else** $\alpha_* = +\infty$;

$\alpha = \min\{\alpha_*, \alpha_F\}$;

if $\alpha = +\infty$ **then stop**; $\qquad$ [the solution is unbounded]

$x \leftarrow x + \alpha p$; $\quad \pi \leftarrow \pi + \alpha q_\pi$; $\quad z_w \leftarrow z_w + \alpha q_w$; $\quad g \leftarrow g + \alpha H p$;

if $\alpha_F < \alpha_*$ **then** $\qquad$ [add constraint r to the working set]

Choose a blocking constraint index r;

Solve $\begin{pmatrix} H & A^T & D_w^T \\ A & 0 & 0 \\ D_w & 0 & 0 \end{pmatrix} \begin{pmatrix} u \\ -v_\pi \\ -v_w \end{pmatrix} = \begin{pmatrix} d_r \\ 0 \\ 0 \end{pmatrix}$;

if $u = 0$ **then** $\sigma = -[z_w]_s/[v_w]_s$ **else** $\sigma = 0$;

$\pi \leftarrow \pi + \sigma v_\pi$; $\quad z_w \leftarrow \begin{pmatrix} z_w + \sigma v_w \\ \sigma \end{pmatrix}$;

$\mathcal{W} \leftarrow \mathcal{W} + \{r\}$; $\quad m_w \leftarrow m_w + 1$;

end;

if $[z_w]_s = 0$ **then** $\qquad$ [delete constraint ν_s from the working set]

$\mathcal{W} \leftarrow \mathcal{W} - \{\nu_s\}$; $\quad m_w \leftarrow m_w - 1$;

for $i = s : m_w$ **do** $[z_w]_i \leftarrow [z_w]_{i+1}$;

$s = \operatorname{argmin}_i\, [z_w]_i$;

end;

$k \leftarrow k + 1$;

end do

Result 2.2.9. *Let x be a subspace minimizer with respect to A_w. Assume the vectors p, q, u and v are defined by* (2.25)*. Let d_r be the gradient of a blocking constraint at $\bar{x} = x + \alpha p$ such that d_r is independent of the columns of A_w^T. If $\rho = -d_r^T p / d_r^T u$, then the vectors*

$$\bar{p} = p + \rho u \quad \textit{and} \quad \bar{q} = \begin{pmatrix} q + \rho v \\ \rho \end{pmatrix}$$

are well-defined and satisfy

$$\begin{pmatrix} H & \bar{A}_w^T \\ \bar{A}_w & \end{pmatrix} \begin{pmatrix} \bar{p} \\ -\bar{q} \end{pmatrix} = \begin{pmatrix} 0 \\ e_{m+s} \end{pmatrix}, \quad \textit{where} \quad \bar{A}_w = \begin{pmatrix} A_w \\ d_r^T \end{pmatrix}. \tag{2.26}$$

Proof. Result 2.2.6 implies that u is nonzero and that $u^T d_r > 0$, so that ρ is well defined (and strictly positive).

For any scalar ρ, (2.25a) and (2.25b) imply that

$$\begin{pmatrix} H & A_w^T & d_r \\ A_w & & \\ d_r^T & & \end{pmatrix} \begin{pmatrix} p + \rho u \\ -(q + \rho v) \\ -\rho \end{pmatrix} = \begin{pmatrix} 0 \\ e_{m+s} \\ d_r^T p + \rho d_r^T u \end{pmatrix}.$$

If ρ is chosen so that $d_r^T p + \rho d_r^T u = 0$, the last component of the right-hand side vanishes, and $\bar{p}$ and $\bar{q}$ satisfy (2.26) as required. $\square$

2.2.3 Relation between the binding and nonbinding methods

Result 2.2.10 (Equivalence of binding and nonbinding directions)**.** *Suppose that x is a standard subspace minimizer with respect to $\mathcal{W}$, and let vectors π and z_w satisfy $g(x) = A^T \pi + D_w^T z_w$. Assume that both the binding- and nonbinding-direction methods identify an index $\nu_s \in \mathcal{W}$ such that $[z_w]_s < 0$. Define the set $\bar{\mathcal{W}} = \mathcal{W} - \{\nu_s\}$.*

Let p be the nonbinding direction from (2.16)*. If the reduced Hessian $Z_{\bar{w}}^T H Z_{\bar{w}}$ is positive definite, then $\bar{p} = \alpha_* p$, where $\bar{p}$ is the binding direction from* (2.6)*, and α_* is the bounded non-binding optimal step $\alpha_* = -[z_w]_s / q_{m+s}$. Otherwise, $\bar{p} = \delta p$, where $\bar{p}$ is defined by* (2.8) *and δ is a bounded positive scalar.*

Proof. Because x is a stationary point, $g(x) = A^T \pi + D_w^T z_w = A^T \pi + D_{\bar{w}}^T z_{\bar{w}} + [z_w]_s d_{\nu_s}$, where $z_{\bar{w}}$ is z_w with the s-th component removed. This implies that

$$[z_w]_s d_{\nu_s} = g(x) - A^T \pi - D_{\bar{w}}^T z_{\bar{w}}. \tag{2.27}$$

By definition, the nonbinding direction p satisfies the equations

$$\begin{pmatrix} H & A_w^T \\ A_w & 0 \end{pmatrix} \begin{pmatrix} p \\ -q \end{pmatrix} = \begin{pmatrix} 0 \\ e_{m+s} \end{pmatrix}.$$

The second block of equations is $A_w p = e_{m+s}$, which implies that $A_{\bar{w}} p = 0$ and $d_{\nu_s}^T p = 1$. Similarly, the first block of equations gives

$$Hp - A_w^T q = Hp - A_{\bar{w}}^T \bar{q} - q_{m+s} d_{\nu_s} = 0, \tag{2.28}$$

where $\bar{q}$ is the $(m + m_{\bar{w}})$-vector defined by removing the $(m+s)$-th component from q.

The definition of the binding direction depends on the inertia of the reduced Hessian $Z_{\bar{w}}^T H Z_{\bar{w}}$. Suppose that it is nonsingular (either positive definite or indefinite). Then $q_{m+s} = p^T H p \neq 0$ since p lies in the null space of $A_{\bar{w}}$ and the binding direction satisfies

$$\begin{pmatrix} H & A_{\bar{w}}^T \\ A_{\bar{w}} & 0 \end{pmatrix} \begin{pmatrix} \bar{p} \\ -\bar{q} \end{pmatrix} = -\begin{pmatrix} g(x) - A^T\pi - D_{\bar{w}}^T z_{\bar{w}} \\ 0 \end{pmatrix}.$$

The equations (2.27) and (2.28) imply that

$$\alpha_*(Hp - A_{\bar{w}}^T \bar{q}) = \alpha_* q_{m+s} d_{\nu_s} = \frac{\alpha_* q_{m+s}}{[z_w]_s} \Big(g(x) - A^T\pi - D_{\bar{w}}^T z_{\bar{w}} \Big) = -(g(x) - A^T\pi - D_{\bar{w}}^T z_{\bar{w}}).$$

Therefore $\alpha_* p$ and $\alpha_* \bar{q}$ satisfy

$$\begin{pmatrix} H & A_{\bar{w}}^T \\ A_{\bar{w}} & 0 \end{pmatrix} \begin{pmatrix} \alpha_* p \\ -\alpha_* \bar{q} \end{pmatrix} = -\begin{pmatrix} g(x) - A^T\pi - D_{\bar{w}}^T z_{\bar{w}} \\ 0 \end{pmatrix}.$$

If $Z_{\bar{w}}^T H Z_{\bar{w}}$ is singular, then $\bar{p}$ and $\bar{q}$ satisfy (2.8)

$$\begin{pmatrix} H & A_{\bar{w}}^T \\ A_{\bar{w}} & 0 \end{pmatrix} \begin{pmatrix} \bar{p} \\ -\bar{q} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

The first equation states that $H\bar{p} - A_{\bar{w}}^T \bar{q} = 0$, which means that $H\bar{p} - A_w^T q = 0$ since $q_{m+s} = p^T H p = 0$ because p lies in the null space of $A_{\bar{w}}$. The second equation implies that $A_w \bar{p} = (d_{\nu_s}^T \bar{p}) e_{m+s}$. If $\delta = 1/d_{\nu_s}^T \bar{p}$, then

$$\begin{pmatrix} H & A_w^T \\ A_w & 0 \end{pmatrix} \begin{pmatrix} \delta \bar{p} \\ -\delta \widehat{q} \end{pmatrix} = \begin{pmatrix} 0 \\ e_{m+s} \end{pmatrix},$$

as required. □

3 Problems in Standard Form

Probably the most common form for expressing quadratic programs, often called *standard form*, is

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\;\; \varphi(x) = c^Tx + \tfrac{1}{2}x^THx \quad \text{subject to} \quad Ax = b, \quad x \geq 0. \tag{3.1}$$

This problem is a particular instance of the mixed constraints $Ax = b$, $Dx \geq f$ in which D is the n-dimensional identity and $f = 0$. The constraints $x \geq 0$, called *simple bounds* or just bounds, are the only inequality constraints in a standard-form problem. Any mixed-constraint problem may be written in standard form. For example, the general inequality constraint $d_i^Tx \geq f_i$ can be converted to a general equality $d_i^Tx - s_i = f_i$ by adding an extra ("slack") variable s_i that is required to be nonnegative. However, QPs in standard form arise naturally in the theory of *duality* (see Chapter 4).

In this chapter, we show that the application of the nonbinding-direction method to a quadratic program in standard-form leads to an algorithm in which the two fundamental systems (2.25a) and (2.25b) may be expressed in terms of a smaller "reduced" KKT system involving a subset of the columns of A.

3.1 Introduction

A first-order KKT point for (3.1) is defined as a point x satisfying the following conditions

$$\begin{aligned}
Ax = b, \qquad x &\geq 0 && \text{(feasibility)}\\
g(x) &= A^T\pi + z && \text{(stationarity)}\\
z &\geq 0 && \text{(nonnegativity)}\\
x \cdot z &= 0 && \text{(complementarity).}
\end{aligned}$$

Since the only inequality constraints of (3.1) are simple bounds on x, the *active set* at a point x is defined as $\mathcal{A}(x) = \{i : x_i = 0\}$, with cardinality $m_{\mathfrak{a}}$. The stationarity and complementarity conditions above are equivalent to the condition $g(x) = A^T\pi + P_{\mathfrak{a}}\, z_{\mathfrak{a}}$, where $z = P_{\mathfrak{a}}\, z_{\mathfrak{a}}$ and $P_{\mathfrak{a}}$ is the $n \times m_{\mathfrak{a}}$ permutation matrix defined by $\mathcal{A}(x)$.

The necessary optimality conditions of (3.1) in active-set format are given in the following result:

Result 3.1.1 (Necessary optimality conditions for standard-form QP). *If x^* is a local minimizer of the quadratic program* (3.1), *then*

(a) $Ax^* = b,\ x \geq 0$;

(b) *there exist vectors π^* and $z_{\mathfrak{a}}$ such that $g(x^*) = A^T\pi^* + P_{\mathfrak{a}}\, z_{\mathfrak{a}}$, where $z_{\mathfrak{a}} \geq 0$ and $P_{\mathfrak{a}}$ is defined by $\mathcal{A}(x^*)$; and*

(b) *it holds that $p^THp \geq 0$ for all nonzero p satisfying $Ap = 0$, and $p_i = 0$ for each $i \in \mathcal{A}(x^*)$.*

3.2 Nonbinding-Direction Method for Standard-Form QP

In standard-form, the working-set matrix D_w consists of rows of the identity matrix, and each working-set index i is associated with a variable x_i that is implicitly fixed at its current value. In this situation, as is customary for constraints in standard form, we refer to the working set as the *nonbasic set* $\mathcal{N}$, and denote its elements as $\{\nu_1, \nu_2, \ldots, \nu_{n_N}\}$ with $n_N = m_w$. The complementary set $\mathcal{B}$ of $n_B = n - n_N$ indices that are not in the working set is known as the *basic set.* The elements of the basic set are denoted by $\{\beta_1, \beta_2, \ldots, \beta_{n_B}\}$.

If P_N denotes the $n \times n_N$ matrix of unit columns $\{\, e_i \,\}$ with $i \in \mathcal{N}$, then the working-set matrix A_w may be written as:

$$A_w = \begin{pmatrix} A \\ P_N^T \end{pmatrix}.$$

Similarly, if P_B is the $n \times n_B$ matrix with unit columns $\{\, e_i \,\}$ with $i \in \mathcal{B}$, then $P = \begin{pmatrix} P_B & P_N \end{pmatrix}$ is a permutation matrix that permutes the columns of A_w as

$$A_w \begin{pmatrix} P_B & P_N \end{pmatrix} = A_w\, P = \begin{pmatrix} A \\ P_N^T \end{pmatrix} P = \begin{pmatrix} AP \\ P_N^T P \end{pmatrix} = \begin{pmatrix} A_B & A_N \\ & I_{n_N} \end{pmatrix},$$

where A_B and A_N are matrices with columns $\{\, a_{\beta_j} \,\}$ and $\{\, a_{\nu_j} \,\}$ respectively. If y is any n-vector, y_B (the *basic components of* y) denotes the n_B-vector whose j-th component is component β_j of y, and y_N (the *nonbasic components of* y) denotes the n_N-vector whose j-th component is component ν_j of y. The same convention is used for matrices, with the exception of I_B and I_N, which are reserved for the identity matrices of order n_B and n_N, respectively. With this notation, the effect of P on the Hessian and working-set matrix may be written as

$$P^THP = \begin{pmatrix} H_B & H_D \\ H_D^T & H_N \end{pmatrix} \quad \text{and} \quad A_w\, P = \begin{pmatrix} A_B & A_N \\ & I_N \end{pmatrix}. \tag{3.2}$$

As in the mixed-constraint formulation, A_w must have full row rank. This is equivalent to requiring that A_B has full row rank since $\operatorname{rank}(A_w) = n_N + \operatorname{rank}(A_B)$.

We will see that for standard-form problems, the nonbinding-direction method is characterized by the basic set instead of the nonbasic (or working) set. Consequently, we redefine a subspace stationary point with respect to a basic set and a second-order-consistent working set as a *second-order-consistent basis.*

Result 3.2.1 (Stationary point and second-order consistent basis)**.** *Let x be a feasible point with basic set $\mathcal{B}$. Let the columns of Z_B form a basis for the null space for A_B.*

(a) *If x is stationary point with respect to A_w, then $g_B = A_B^T\pi$ for some vector π, or equivalently, the reduced gradient $Z_B^T g_B = 0$ and x is referred to as a* subspace stationary point with respect to $\mathcal{B}$ *(or A_B).*

(b) *If $\mathcal{B}$ is a* second-order-consistent basis *for* (3.1)*, then the reduced Hessian $Z_B^T H Z_B$ is positive definite. Equivalently, the KKT matrix $K_B = \begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix}$ has inertia $(n_B, m, 0)$.*

Proof. Definition 2.2.1 implies that there exists a vector y such that $g(x) = A_w^T y$. Applying the permutation P to the equation implies

$$\begin{pmatrix} g_B \\ g_N \end{pmatrix} = P^T g = P^T A_w^T y = \begin{pmatrix} A_B^T & \\ A_N^T & I_N \end{pmatrix} y,$$

so that $g_B = A_B^T \pi \in \mathrm{range}(A_B^T)$, where the vector π is the first m components of the vector y.

For part (b), let the columns of Z define a basis for the null space of A_w. Applying the permutation P^T of (3.2) to Z gives

$$P^T Z = \begin{pmatrix} Z_B \\ Z_N \end{pmatrix}.$$

Then

$$A_w Z = A_w P P^T Z = \begin{pmatrix} A_B & A_N \\ & I_N \end{pmatrix} \begin{pmatrix} Z_B \\ Z_N \end{pmatrix} = \begin{pmatrix} A_B Z_B + A_N Z_N \\ Z_N \end{pmatrix} = 0,$$

so that $Z_N = 0$. This implies that

$$Z^T H Z = Z^T P P^T H P P^T Z = \begin{pmatrix} Z_B^T & Z_N^T \end{pmatrix} \begin{pmatrix} H_B & H_D \\ H_D^T & H_N \end{pmatrix} \begin{pmatrix} Z_B \\ Z_N \end{pmatrix} = Z_B^T H_B Z_B.$$

Consequently, $Z^T H Z$ is positive definite if and only if $Z_B^T H_B Z_B$ is positive definite. Moreover, $\mathrm{In}(Z^T H Z) = \mathrm{In}(Z_B^T H_B Z_B)$.

By definition, since x is a subspace minimizer, $Z^T H Z$ is positive definite and has inertia $(n - (m + n_N), 0, 0)$. By Corollary 1.3.1, the inertia of K_B satisfies

$$\begin{aligned} \mathrm{In}(K_B) = \mathrm{In}(Z_B^T H Z_B) + (m, m, 0) &= \mathrm{In}(Z^T H Z) + (m, m, 0) \\ &= (n - (m + n_N), 0, 0) + (m, m, 0) \\ &= (n - n_N, m, 0) = (n_B, m, 0). \end{aligned}$$

□

As in linear programming, the components of the vector $z = g(x) - A^T\pi$ are called the *reduced costs*. For constraints in standard form, the multipliers z_w associated inequality constraints in the working set are denoted by z_N, whose components are the nonbasic components of the reduced-cost vector, i.e.,

$$z_N = (g(x) - A^T\pi)_{\mathcal{N}} = g_N - A_N^T\pi.$$

At a subspace stationary point, it holds that $g_B - A_B^T\pi = 0$, which implies that the basic components of the reduced costs z_B are zero.

The fundamental property of constraints in standard form is that the mixed-constraint method may be formulated so that the number of variables involved in the equality-constraint QP subproblem (2.15) is reduced from n to n_B. Suppose that $z_{\nu_s} < 0$ for $\nu_s \in \mathcal{N}$. By applying the permutation matrix P to the KKT system (2.25a), we have

$$\left(\begin{array}{cc|cc} H_B & H_D & A_B^T & \\ H_D^T & H_N & A_N^T & I_N \\ \hline A_B & A_N & & \\ & I_N & & \end{array}\right) \begin{pmatrix} p_B \\ p_N \\ -q_\pi \\ -q_N \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ e_s \end{pmatrix}, \quad \text{where } p = P\begin{pmatrix} p_B \\ p_N \end{pmatrix} \text{ and } q = \begin{pmatrix} q_\pi \\ q_N \end{pmatrix}. \tag{3.3}$$

These equations imply that $p_N = e_s$ and p_B and q_π satisfy the reduced KKT system

$$\begin{pmatrix} H_B & A_B^T \\ A_B & 0 \end{pmatrix} \begin{pmatrix} p_B \\ -q_\pi \end{pmatrix} = \begin{pmatrix} -H_D p_N \\ -A_N p_N \end{pmatrix} = -\begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} \\ a_{\nu_s} \end{pmatrix}. \tag{3.4}$$

In practice, p_N is defined implicitly and only the components of p_B and q_π are computed explicitly. Once p_B and q_π are known, the increment q_N for multipliers z_N associated with the constraints $p_N = e_s$ are given by $q_N = (Hp - A^Tq_\pi)_{\mathcal{N}}$. The computed search directions satisfy the identities in Result 2.2.4. In terms of the standard form variables, these identities imply

$$g^Tp = [z_N]_s \text{ and } p^THp = [q_N]_s, \tag{3.5}$$

so that the optimal step $\alpha_* = -[z_N]_s/[q_N]_s$.

The solution of the second KKT system (2.25b) can be similarly computed from the KKT equation

$$\begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix} \begin{pmatrix} u_B \\ -v_\pi \end{pmatrix} = \begin{pmatrix} e_r \\ 0 \end{pmatrix}, \tag{3.6}$$

with $u_N = 0$ and $v_N = (Hu - A^Tv_\pi)_{\mathcal{N}}$, where $u = P\begin{pmatrix} u_B \\ u_N \end{pmatrix}$ and $v = \begin{pmatrix} v_\pi \\ v_N \end{pmatrix}$.

The KKT equations (3.4) and (3.6) allow the mixed-constraint algorithm to be formulated in terms of the basic variables only, which implies that the algorithm is driven by variables entering or leaving the basic set rather than constraints entering or leaving the working set. With this interpretation, changes to the KKT matrix are based on column-changes to A_B instead of row-changes to A_w.

For completeness Results 2.2.5—2.2.8 are summarized in terms of the quantities associated with constraints in standard form.

Result 3.2.2. *Let x be a subspace minimizer with respect to the basic set $\mathcal{B}$, with $[z_N]_s < 0$. Let $\bar{x}$ be the point such that $\bar{x}_N = x_N + \alpha e_s$ and $\bar{x}_B = x_B + \alpha p_B$, where p_B is defined as in (3.4).*

(1) *The step to the minimizer of $\varphi(x + \alpha p)$ is $\alpha_* = -z_{\nu_s}/[q_N]_s$. If α_* is bounded and $\alpha = \alpha_*$, then $\bar{x}$ is a subspace minimizer with respect to the basic set $\bar{\mathcal{B}} = \mathcal{B} + \{\nu_s\}$.*

(2) *Alternatively, the largest feasible step is defined by the minimum ratio test:*

$$\alpha_F = \min \gamma_i, \quad \text{where} \quad \gamma_i = \begin{cases} \dfrac{[x_B]_i}{-[p_B]_i} & \text{if } [p_B]_i < 0, \\ +\infty & \text{otherwise.} \end{cases} \tag{3.7}$$

Suppose $\alpha = \alpha_F$ and $[x_B + \alpha p_B]_{\beta_r} = 0$ and let u_B and v_π be defined by (3.6).

(a) *e_r and the columns of A_B^T are linearly independent if and only if $u_B \neq 0$.*

(b) *$[v_N]_s = -[p_B]_r > 0$, and if $u_B \neq 0$, then $[u_B]_r > 0$.*

(c) *If e_r and the columns of A_B^T are linearly independent, then $\bar{x}$ is a subspace minimizer with respect to $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\}$. Moreover, $g_{\bar{B}}(\bar{x}) = A_{\bar{B}}^T \bar{\pi}$ and $g_{\bar{N}}(\bar{x}) = A_{\bar{N}}^T \bar{\pi} + \bar{z}_N$, where $\bar{\pi} = \pi + \alpha q_\pi$ and $\bar{z}_N$ is formed by appending a zero component to the vector $z_N + \alpha q_N$.*

(d) *If e_r and the columns of A_B^T are linearly dependent, define $\sigma = -[z_N + \alpha q_N]_s/[v_N]_s$. Then $\bar{x}$ is a subspace minimizer with respect to $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\} + \{\nu_s\}$ with $g_{\bar{B}}(\bar{x}) = A_{\bar{B}}^T \bar{\pi}$ and $g_{\bar{N}}(\bar{x}) = A_{\bar{N}}^T \bar{\pi} + \bar{z}_N$, where $\bar{\pi} = \pi + \alpha q_\pi + \sigma v_\pi$ with $\sigma > 0$, and $\bar{z}_N$ is formed by appending σ to $z_N + \alpha q_N + \sigma v_N$.*

Proof. For part (1), we first show that $\bar{x}$ remains a stationary point for $\bar{\mathcal{B}}$. Since $\alpha = \alpha_* = -[z_N]_s/[q_N]_s$, the multiplier of the ν_s-th constraint $[z_N + \alpha q_N]_s = 0$ so that $z_{\bar{B}} = 0$.

Now let K_B and $K_{\bar{B}}$ denote the matrices associated with basic sets $\mathcal{B}$ and $\bar{\mathcal{B}}$. We must show that $K_{\bar{B}}$ has the correct inertia. However, since inertia is unchanged by symmetric permutations, we consider a permuted version of $K_{\bar{B}}$:

$$\widetilde{K}_{\bar{B}} = Q^T K_{\bar{B}} Q = \left(\begin{array}{cc|c} H_B & A_B^T & (h_{\nu_s})_{\mathcal{B}} \\ A_B & & a_{\nu_s} \\ \hline (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T & h_{\nu_s,\nu_s} \end{array}\right)$$

where Q is a permutation matrix. Because K_B is associated with a subspace minimizer, K_B is nonsingular with $\text{In}(K_B) = (n_B, m, 0)$. In particular, $\widetilde{K}_{\bar{B}}/K_B$ the Schur complement of K_B in $\widetilde{K}_{\bar{B}}$ exists with

$$\widetilde{K}_{\bar{B}}/K_B = h_{\nu_s,\nu_s} - \begin{pmatrix} (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T \end{pmatrix} K_B^{-1} \begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} \\ a_{\nu_s} \end{pmatrix}.$$

It follows from equation (3.4) that

$$K_B \begin{pmatrix} p_B \\ -q_\pi \end{pmatrix} = -\begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} \\ a_{\nu_s} \end{pmatrix}.$$

Thus, the Schur complement can be written as

$$\begin{aligned} \widetilde{K}_{\bar{B}}/K_B &= h_{\nu_s,\nu_s} - \begin{pmatrix} (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T \end{pmatrix} \begin{pmatrix} -p_B \\ q_\pi \end{pmatrix} \\ &= h_{\nu_s,\nu_s} + (h_{\nu_s})_{\mathcal{B}}^T p_B - a_{\nu_s}^T q_\pi \\ &= e_s^T H_N e_s + e_s^T H_D^T p_B - e_s^T A_N^T q_\pi \\ &= e_s^T q_N = [q_N]_s \quad \text{by (3.4)} \ . \end{aligned}$$

Then $\mathrm{In}(K_{\bar{B}}) = \mathrm{In}(\widetilde{K}_{\bar{B}}) = \mathrm{In}(K_B) + \mathrm{In}(\widetilde{K}_{\bar{B}}/K_B) = \mathrm{In}(K_B) + \mathrm{In}([q_N]_s)$.

Since α_* is bounded, $[q_N]_s = p^T H p$ must be positive, so that $\mathrm{In}([q_N]_s) = (1, 0, 0)$. It follows the KKT matrix associated with $\bar{\mathcal{B}}$ has inertia $(n_B + 1, m, 0)$ and the subspace stationary point $\bar{x}$ is a subspace minimizer with respect to $\bar{\mathcal{B}}$.

For part (2a), equation (3.6) implies that $H_B u_B - A_B^T v_\pi = e_r$ and $A_B u_B = 0$. If $u_B = 0$, then $-A_B^T v_\pi = e_r$ so e_r must be dependent on the rows of A_B. Conversely, if $-A_B^T v_\pi = e_r$, then the definition of u_B gives $u_B^T e_r = -u_B^T A_B^T v_\pi = 0$, which implies $u_B^T H_B u_B = 0$. By assumption, x is a subspace minimizer with respect to $\mathcal{B}$ which is equivalent to the assumption that H_B is positive definite for all u_B such that $A_B u_B = 0$. Thus, $u_B^T H_B u_B = 0$ can hold only if $u_B = 0$.

Part (2b) follows directly from Result 2.2.6 since $v_{m+s} = [v_N]_s = -e_{\beta_r}^T p = -[p_B]_r > 0$ and $u^T e_{\beta_r} = [u_B]_r > 0$ if $u_B \neq 0$.

For part (2c), observe that (2.19) implies

$$g_B(\bar{x}) = A_B^T(\pi + \alpha q_\pi) \quad \text{and} \quad g_N(\bar{x}) = A_N^T(\pi + \alpha q_\pi) + (z_N + \alpha q_N).$$

Since e_r and the rows of A_B are linearly independent, the index β_r may be added to the nonbasic set. The new basic and nonbasic sets are defined as $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\}$ and $\bar{\mathcal{N}} = \mathcal{N} + \{\beta_r\}$. The column of A_B corresponding to the t-th variable is removed from A_B to form $A_{\bar{B}}$ and is appended to A_N to form the new nonbasic matrix $A_{\bar{N}}$. Then

$$g_{\bar{B}}(\bar{x}) = A_{\bar{B}}^T \bar{\pi} \quad \text{and} \quad g_{\bar{N}}(\bar{x}) = A_{\bar{N}}^T \bar{\pi} + \bar{z}_N,$$

where $\bar{z}_N$ is formed by appending a zero to the vector $z_N + \alpha q_N$.

It suffices to show that for $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\}$, $K_{\bar{B}}$ has inertia $(n_B - 1, m, 0)$. Consider the matrix

$$M \triangleq \begin{pmatrix} K_B & e_r \\ e_r^T & \end{pmatrix}.$$

By assumption, x is a subspace minimizer and $\operatorname{In}(K_B) = (n_B, m, 0)$. Thus, K_B is nonsingular and the Schur complement of K_B in M exists with

$$\begin{aligned} M/K_B = -e_r^T K_B^{-1} e_r = -e_r^T \begin{pmatrix} u_B \\ -v_\pi \end{pmatrix} \quad \text{by (3.6)} \\ = -[u_B]_r < 0. \end{aligned}$$

Then,

$$\begin{aligned} \operatorname{In}(M) = \operatorname{In}(M/K_B) + \operatorname{In}(K_B) &= \operatorname{In}(-[u_B]_r) + (n_B, m, 0) \\ &= (0,1,0) + (n_B, m, 0) \\ &= (n_B, m+1, 0). \end{aligned} \tag{3.8}$$

Since $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\}$, a permutation can be applied to K_B such that

$$K_B = \begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix} \sim \left(\begin{array}{cc|c} H_{\bar{B}} & (h_{\beta_r})_{\bar{B}} & A_{\bar{B}}^T \\ (h_{\beta_r})_{\bar{B}}^T & h_{\beta_r,\beta_r} & a_{\beta_r}^T \\ \hline A_{\bar{B}} & a_{\beta_r} & 0 \end{array}\right).$$

Similarly, applying symmetric permutations to M gives

$$\begin{aligned} M \triangleq \begin{pmatrix} K_B & e_r \\ e_r^T & \end{pmatrix} &\sim \left(\begin{array}{ccc|c} H_{\bar{B}} & (h_{\beta_r})_{\bar{B}} & A_{\bar{B}}^T & 0 \\ (h_{\beta_r})_{\bar{B}}^T & h_{\beta_r,\beta_r} & a_{\beta_r}^T & 1 \\ A_{\bar{B}} & a_{\beta_r} & 0 & 0 \\ \hline 0 & 1 & 0 & 0 \end{array}\right) \\ &\sim \left(\begin{array}{cc|cc} h_{\beta_r,\beta_r} & 1 & (h_{\beta_r})_{\bar{B}}^T & a_{\beta_r}^T \\ 1 & 0 & 0 & 0 \\ \hline (h_{\beta_r})_{\bar{B}} & 0 & H_{\bar{B}} & A_{\bar{B}}^T \\ a_{\beta_r} & 0 & A_{\bar{B}} & 0 \end{array}\right) \triangleq \widetilde{M}. \end{aligned}$$

The leading 2×2 block of $\widetilde{M}$, denoted by E, has $\det(E) = -1$ so $\operatorname{In}(E) = (1,1,0)$. The Schur complement of E in $\widetilde{M}$ is

$$\begin{aligned} \widetilde{M}/E &= K_{\bar{B}} - \begin{pmatrix} (h_{\beta_r})_{\bar{B}} & 0 \\ a_{\beta_r} & 0 \end{pmatrix} \begin{pmatrix} h_{\beta_r,\beta_r} & 1 \\ 1 & 0 \end{pmatrix}^{-1} \begin{pmatrix} (h_{\beta_r})_{\bar{B}}^T & a_{\beta_r}^T \\ 0 & 0 \end{pmatrix} \\ &= K_{\bar{B}} - \begin{pmatrix} (h_{\beta_r})_{\bar{B}} & 0 \\ a_{\beta_r} & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -h_{\beta_r,\beta_r} \end{pmatrix} \begin{pmatrix} (h_{\beta_r})_{\bar{B}}^T & a_{\beta_r}^T \\ 0 & 0 \end{pmatrix} \\ &= K_{\bar{B}}, \end{aligned}$$

which implies that $\operatorname{In}(M) = \operatorname{In}(\widetilde{M}) = \operatorname{In}(\widetilde{M}/E) + \operatorname{In}(E) = \operatorname{In}(K_{\bar{B}}) + (1,1,0)$. Combining this with (3.8) yields

$$\operatorname{In}(K_{\bar{B}}) = \operatorname{In}(M) - (1,1,0) = (n_B, m+1, 0) - (1,1,0) = (n_B - 1, m, 0),$$

so that $K_{\bar{B}}$ has correct inertia and $\bar{x}$ is a subspace minimizer with respect to $\bar{\mathcal{B}}$.

For part (2d), assume that e_r and the rows of A_B are linearly dependent so that $u_B = 0$ with $-A_B^T v_\pi = e_r$ and $v_N = -A_N^T v_\pi$.

Let σ be an arbitrary scalar. It follows that the basic components of the gradient satisfy

$$\begin{aligned} g_B(\bar{x}) = A_B^T(\pi + \alpha q_\pi) &= A_B^T(\pi + \alpha q_\pi + \sigma v_\pi) - \sigma A_B^T v_\pi \\ &= A_B^T(\pi + \alpha q_\pi + \sigma v_\pi) + \sigma e_r \\ &= A_B^T \bar{\pi} + \sigma e_r, \end{aligned}$$

where $\bar{\pi} = \pi + \alpha q_\pi + \sigma v_\pi$. Similarly, for the nonbasic components, it follows that

$$\begin{aligned} g_N(\bar{x}) &= A_N^T(\pi + \alpha q_\pi) + z_N + \alpha q_N \\ &= A_N^T(\pi + \alpha q_\pi + \sigma v_\pi) + z_N + \alpha q_N - \sigma A_N^T v_\pi \\ &= A_N^T \bar{\pi} + \bar{z}_N, \quad \text{with} \;\; \bar{z}_N = z_N + \alpha q_N + \sigma v_N. \end{aligned}$$

If σ is defined as $\sigma = -[z_N + \alpha q_N]_s / [v_N]_s$, then $[\bar{z}_N]_s = [z_N + \alpha q_N - \sigma v_N]_s = 0$. This implies that the next basic and nonbasic sets can be defined as $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\} + \{\nu_s\}$ and $\bar{\mathcal{N}} = \mathcal{N} + \{\beta_r\} - \{\nu_s\}$, so that

$$g_{\bar{B}}(\bar{x}) = A_{\bar{B}}^T \bar{\pi} \quad \text{and} \quad g_{\bar{N}}(\bar{x}) = A_{\bar{N}}^T \bar{\pi} + \widetilde{z}_N,$$

with $\bar{\pi} = \pi + \alpha q_\pi + \sigma v_\pi$ and $\widetilde{z}_N$ formed by appending σ to $\bar{z}_N$.

To show that $\sigma > 0$, notice that $\eta(\alpha) = [z_N + \alpha q_N]_s$ is a nondecreasing linear function of α such that $\eta(0) = [z_N]_s < 0$ and $\eta(\alpha_*) = 0$. This implies that if a constraint is blocking, then $\alpha < \alpha_*$ and $[z_N + \alpha q_N]_s < 0$. Now $\sigma > 0$ if $[v_N]_s > 0$. But $[v_N]_s = e_s^T(-A_N^T v_\pi) = -a_{\nu_s}^T v_\pi = p_B^T A_B^T v_\pi = -p_B^T e_r = -[p_B]_r > 0$ since r is the index of a blocking constraint. Thus $\sigma > 0$.

Let K_B and $K_{\bar{B}}$ denote the KKT matrices associated with $\mathcal{B}$ and $\bar{\mathcal{B}}$ and denote the intermediate basic set $\mathcal{B} - \{\beta_r\}$ as $\widehat{\mathcal{B}}$. Since $\bar{\mathcal{B}}$ is $\mathcal{B}$ with the r-th index replaced by ν_s, $K_{\bar{B}}$ differs from K_B by a single row and column. Although it is very similar to the proofs in part (1) and (2c), a concise proof to show that $K_{\bar{B}}$ has correct inertia is provided for completeness.

Define the matrix M as

$$\left(\begin{array}{cc|cc} H_B & A_B^T & (h_{\nu_s})_{\mathcal{B}} & e_r \\ A_B & 0 & a_{\nu_s} & 0 \\ \hline (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T & h_{\nu_s,\nu_s} & 0 \\ e_r^T & 0 & 0 & 0 \end{array}\right).$$

The $(1,1)$-block is K_B, which is nonsingular, so that the Schur complement M/K_B is

$$\begin{aligned} M/K_B &= \begin{pmatrix} h_{\nu_s,\nu_s} & 0 \\ 0 & 0 \end{pmatrix} - \begin{pmatrix} (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T \\ e_r^T & 0 \end{pmatrix} K_B^{-1} \begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} & e_r \\ a_{\nu_s} & 0 \end{pmatrix} \\ &= \begin{pmatrix} h_{\nu_s,\nu_s} & 0 \\ 0 & 0 \end{pmatrix} - \begin{pmatrix} (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T \\ e_r^T & 0 \end{pmatrix} \begin{pmatrix} -p_B & 0 \\ q_\pi & -v_\pi \end{pmatrix} \\ &= \begin{pmatrix} h_{\nu_s,\nu_s} + (h_{\nu_s})_{\mathcal{B}}^T p_B - a_{\nu_s}^T q_\pi & a_{\nu_s}^T v_\pi \\ e_r^T p_B & 0 \end{pmatrix} \\ &= \begin{pmatrix} [q_N]_s & [v_N]_s \\ [p_B]_r & 0 \end{pmatrix} \end{aligned}$$

Since $[v_N]_s = -[p_B]_r > 0$, M/K_B has inertia $(1,1,0)$. Thus, $\mathrm{In}(M) = (n_B, m, 0) + (1,1,0) = (n_B+1, m+1, 0)$.

Now consider a permuted M such that

$$\bar{M} = \left(\begin{array}{ccc|cc} H_{\widehat{B}} & A_{\widehat{B}}^T & (h_{\nu_s})_{\widehat{\mathcal{B}}} & (h_{\beta_r})_{\widehat{\mathcal{B}}} & 0 \\ A_{\widehat{B}} & 0 & a_{\nu_s} & a_{\beta_r} & 0 \\ (h_{\nu_s})_{\widehat{\mathcal{B}}}^T & a_{\nu_s}^T & h_{\nu_s,\nu_s} & 0 & 0 \\ \hline (h_{\beta_r})_{\widehat{\mathcal{B}}}^T & a_{\beta_r}^T & 0 & h_{\beta_r,\beta_r} & 1 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right).$$

Since the $(1,1)$-block of this matrix is a permuted version of $K_{\bar{B}}$, it remains to show that this block has correct inertia. Notice that the $(2,2)$-block of the above matrix (which we denote by E) is nonsingular, so that the Schur complement must exist. By a simple calculation, $\bar{M}/E = K_{\bar{B}}$. Therefore $\mathrm{In}(K_{\bar{B}}) = \mathrm{In}(\bar{M}) - \mathrm{In}(E) = (n_B, m, 0)$. □

As in the general mixed-constraint method, the direction p_B and multiplier q_π can be updated in the linearly independent case.

Result 3.2.3. *Let x be a subspace minimizer with respect to $\mathcal{B}$. Assume the vectors p_B, q_π, u_B and v_π are defined by* (3.4) *and* (3.6)*. Let β_r be the index of a linearly independent blocking constraint at $\bar{x}$, where $\bar{x}_N = x_N + \alpha e_s$ and $\bar{x}_B = x_B + \alpha p_B$. Let $\rho = -[p_B]_r/[u_B]_r$, and consider the vectors $\bar{p}_B$ and $\bar{q}_\pi$, where $\bar{p}_B$ is the vector $p_B + \rho u_B$ with the r-th component omitted, and $\bar{q}_\pi = q_\pi + \rho v_\pi$. Then $\bar{p}_B$ and $\bar{q}_\pi$ are well-defined and satisfy the KKT equations for the basic set $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\}$.*

Proof. Since the blocking constraint is linearly independent, $u_B \neq 0$ and $[u_B]_r$ is nonzero by part (2b) of Result 3.2.2, so that ρ is well-defined.

Let K be the matrix K_B with the r-th components zeroed out, i.e., $K = K_B - K_r$, where

$$K_r = \begin{pmatrix} H_B e_r e_r^T + e_r (H_B e_r)^T - h_{\beta_r\beta_r} e_r e_r^T & e_r e_r^T A_B^T \\ A_B e_r e_r^T & \end{pmatrix}.$$

$$
\begin{array}{llcccccc}
 & x_0 & \longrightarrow & \cdots\ x_{k-1} & \longrightarrow & x_k & \longrightarrow & x_{k+1} \\[2ex]
\text{(A)} & \mathcal{B}_0 & \overset{\textit{move, delete}}{\longrightarrow} & \cdots\ \mathcal{B}_{k-1} & \overset{\textit{move, delete}}{\longrightarrow} & \mathcal{B}_k & \overset{\textit{move, add } \nu_s}{\longrightarrow} & \mathcal{B}_{k+1} \\[2ex]
\text{(B)} & \mathcal{B}_0 & \overset{\textit{move, delete}}{\longrightarrow} & \cdots\ \mathcal{B}_{k-1} & \overset{\textit{move, delete}}{\longrightarrow} & \mathcal{B}_k & \overset{\textit{move, swap } \nu_s \text{ \& } \beta_r}{\longrightarrow} & \mathcal{B}_{k+1}
\end{array}
$$

Figure 3.1: This figure depicts the two types of sequence of consecutive iterates in the nonbinding-direction method. Each sequence starts and ends with standard subspace minimizers x_0 and x_{k+1}. Intermediate iterates are nonstandard subspace minimizers. The sequences differ in how the final point is reached. In (A), ν_s is added to the basic set after an optimal step $\alpha = \alpha_*$. In (B), β_r is the index of a linearly dependent blocking constraint and it is swapped with the ν_s-th constraint after a blocking step ($\alpha_F < \alpha_*$) is taken.

Then

$$K_B \begin{pmatrix} p_B + \rho u_B \\ -(q_\pi + \rho v_\pi) \end{pmatrix} = -\begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} \\ a_{\nu_s} \end{pmatrix} + \rho \begin{pmatrix} e_r \\ 0 \end{pmatrix} \quad \text{and} \quad K_r \begin{pmatrix} p_B + \rho u_B \\ -(q_\pi - \rho v_\pi) \end{pmatrix} = \begin{pmatrix} \rho e_r - [h_{\nu_s}]_r e_r \\ 0 \end{pmatrix},$$

so that

$$K \begin{pmatrix} p_B + \rho u_B \\ -(q_\pi - \rho v_\pi) \end{pmatrix} = -\begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} - [h_{\nu_s}]_r e_r \\ a_{\nu_s} \end{pmatrix}.$$

If $\bar{p}_B$ is the vector $p_B + \rho u_B$ with the r-th component removed, then the above equation implies that

$$K_{\bar{B}} \begin{pmatrix} \bar{p}_B \\ -\bar{q}_\pi \end{pmatrix} = -\begin{pmatrix} (h_{\nu_s})_{\bar{\mathcal{B}}} \\ a_{\nu_s} \end{pmatrix},$$

where $K_{\bar{B}}$ is the KKT matrix associated with $\bar{\mathcal{B}}$. $\square$

The standard-form version of the nonbinding-direction method computes sequences of iterates that start and end with a standard subspace minimizer with intermediate iterates consisting of nonstandard subspace minimizers. Figure 3.1 shows the two possible types of sequences. In both sequences, intermediate iterates are reached by taking blocking steps where the blocking constraint is linearly independent of the constraints in the current basic set. In the upper sequence (A), the final standard subspace minimizer is reached when an optimal step is taken and ν_s is added to the basic set. In the lower sequence (B), we encounter a blocking constraint that is linearly dependent of the basic set constraints. In this case, ν_s is added to the basic set and the index β_r of the blocking constraint is removed.

Algorithm 3.1 summarizes the nonbinding-direction method for quadratic programming. Instead of using the vectors q_N and v_N to update z, the algorithm recomputes z from π using $z = g - A^T\pi$. Furthermore, the relation in part 2(b) of Result 3.2.2 is used to simplify the computation of $[v_N]_s$.

Algorithm 3.1: Nonbinding-direction method for a general QP in standard form

Find x_0 such that $Ax_0 = b$ and $x_0 \geq 0$;

$[x, \pi, \mathcal{B}, \mathcal{N}] = \texttt{subspaceMin}(x_0)$;

$g = c + Hx$; $z = g - A^T\pi$;

$\nu_s = \text{argmin}_i\{z_i\}$;

while $z_{\nu_s} < 0$ **do**

  Solve $\begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix} \begin{pmatrix} p_B \\ -q_\pi \end{pmatrix} = -\begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} \\ a_{\nu_s} \end{pmatrix}$; $p_N = e_s$; $p = P\begin{pmatrix} p_B \\ p_N \end{pmatrix}$;

  $\alpha_F = \texttt{minRatioTest}(x_B, p_B)$;

  if $[q_N]_s > 0$ **then** $\alpha_* = -z_{\nu_s}/[q_N]_s$ **else** $\alpha_* = +\infty$;

  $\alpha = \min\{\alpha_*, \alpha_F\}$;

  if $\alpha = +\infty$ **then stop**;   [the solution is unbounded]

  $x \leftarrow x + \alpha p$; $g \leftarrow g + \alpha Hp$;

  $\pi \leftarrow \pi + \alpha q_\pi$; $z = g - A^T\pi$;

  if $\alpha_F < \alpha_*$ **then**   [remove the r-th basic variable]

    Find the blocking constraint index r;

    Solve $\begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix} \begin{pmatrix} u_B \\ -v_\pi \end{pmatrix} = \begin{pmatrix} e_r \\ 0 \end{pmatrix}$;

    if $u_B = 0$ **then** $\sigma = z_{\nu_s}/[p_B]_r$ **else** $\sigma = 0$;

    $\mathcal{B} \leftarrow \mathcal{B} - \{\beta_r\}$; $\mathcal{N} \leftarrow \mathcal{N} + \{\beta_r\}$;

    $\pi \leftarrow \pi + \sigma v_\pi$; $z = g - A^T\pi$;

  end;

  if $z_{\nu_s} = 0$ **then**   [add the s-th nonbasic variable]

    $\mathcal{B} \leftarrow \mathcal{B} + \{\nu_s\}$; $\mathcal{N} \leftarrow \mathcal{N} - \{\nu_s\}$;

    $\nu_s = \text{argmin}_i\{z_i\}$;

  end;

  $k \leftarrow k + 1$;

end do

3.3 Linear Programs in Standard Form

If the problem is a linear program (i.e., $H = 0$), then the basic set $\mathcal{B}$ may be chosen so that A_B is always nonsingular (i.e., it is square with rank m). In this case, we show that Algorithm 3.1 simplifies to a variant of the primal simplex method in which the π-values and reduced costs are updated by a two-term recurrence relation.

When $H = 0$, the equations (3.4) reduce to $A_B p_B = -a_{\nu_s}$ and $A_B^T q_\pi = 0$, with $p_N = e_s$ and $q_N = -A_N^T q_\pi$. Since A_B is nonsingular, both q_π and q_N are zero, and the directions p_B and p_N are identical to those defined by the simplex method. In the case of (3.6), the basic and nonbasic components of u satisfy $A_B u_B = 0$ and $u_N = 0$. Similarly, $v_N = -A_N^T v_\pi$, where $-A_B^T v_\pi = e_r$. Again, as A_B is nonsingular, $u_B = 0$ and the linearly dependent case always applies in Algorithm 3.1. This implies that the r-th basic and the s-th nonbasic variables are always swapped, as in the primal simplex method. Every iterate for an LP is a standard subspace minimizer.

As q_π and q_N are zero, the updates to the multiplier vectors π and z_N defined by part 2(d) of Result 3.2.2 depend only on v_π, v_N and the scalar $\sigma = -[z_N]_s/[v_N]_s$. The resulting updates to the multipliers are:

$$\pi \leftarrow \pi + \sigma v_\pi, \quad \text{and} \quad z_N \leftarrow \begin{pmatrix} z_N + \sigma v_N \\ \sigma \end{pmatrix},$$

which are used in many implementations of the simplex method.

4 Dual Quadratic Programming

In this chapter, we formulate a dual active-set method by applying the nonbinding-direction method to the *dual problem* of the standard-form quadratic problem introduced in Chapter 3. The original "primal" standard-form problem is restated here:

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; \varphi(x) = c^Tx + \tfrac{1}{2}x^THx \quad \text{subject to} \quad Ax = b, \quad x \geq 0. \tag{4.1}$$

The stationarity condition of the primal QP gives an explicit relation between the primal variables x and the dual variable π and z. Based on this condition, a dual problem is formulated where the roles of the primal and dual variables are reversed. Instead of minimizing over the primal variables x, a dual QP minimizes over variables π and z that satisfy the stationarity and non-negativity conditions of the primal QP. If the original primal problem is not convex, it may not be possible to recover a primal solution from the dual. Therefore, the dual method is only applied to convex primal problems, i.e., to problems with positive-semidefinite H.

The relationship between the primal and dual was first given by Dorn in [21]. A dual active-set method for strictly convex problems was proposed by Goldfarb and Idnani [47]. This method was extended by Powell [60] to deal with ill-conditioned problems, and reformulated by Boland [5] to handle the general convex case. These methods require the factorization of a matrix defined in terms of the inverse of H, and as such, they are unsuitable for large-scale QP. In particular, the Goldfarb-Idnani method uses a range-space method to solve a KKT system of the form

$$\begin{pmatrix} H & A_{\mathfrak{a}}^T \\ A_{\mathfrak{a}} & 0 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} a_j \\ 0 \end{pmatrix}.$$

The solution is defined by the inverse of the Hessian and the Moore-Penrose pseudoinverse such that

$$M^\dagger = (A_{\mathfrak{a}} H^{-1} A_{\mathfrak{a}}^T)^{-1} A_{\mathfrak{a}} H^{-1} \;\text{ and }\; N = H^{-1}(I - A_{\mathfrak{a}}^T M^\dagger),$$

with $p = Na_j$ and $q = M^\dagger a_j$. The pseudoinverse M and matrix N are not computed explicitly, but are stored in factored form as dense matrices. The difficulty of using the inverse of H and dense factorizations was addressed by Bartlett and Biegler [2] in the code QPSchur, which is a reformulation of the Goldfarb-Idnani method utilizing the Schur-complement method to solve the linear systems (see Section 7.2 for a discussion of the Schur-complement method). However, QPSchur is only appropriate for strictly convex problems as strict convexity is required to ensure a positive definite reduced Hessian at every iteration of the method.

In the next section, background information on dual problems is given and the dual problem format is introduced. In Section 4.2, the dual version of the nonbinding algorithm is described.

4.1 Background

A point x satisfying the constraints of the primal problem is called *primal feasible.* Multipliers π and z satisfying the stationarity and non-negativity conditions (i.e., $g(x) = A^T\pi + z$ and $z \geq 0$) of (4.1) are called *dual feasible.* Given such primal-dual points, we have

$$0 \leq z^T x = (c + Hx - A^T\pi)^T x = c^T x + \tfrac{1}{2}x^T Hx + \tfrac{1}{2}x^T Hx - b^T\pi,$$

which implies that $\varphi(x) \geq -(\frac{1}{2}x^T Hx - b^T\pi)$. Based on this inequality, we wish to determine π and z by maximizing $-\frac{1}{2}x^T Hx + b^T\pi$ or, equivalently, minimizing the dual quadratic objective function $\varphi_D(x,\pi) = \frac{1}{2}x^T Hx - b^T\pi$ over the set of dual feasible points.

The "dual" quadratic problem for (4.1) is written as

$$\begin{array}{ll} \underset{w,z\in\mathbb{R}^n,\pi\in\mathbb{R}^m}{\text{minimize}} & \varphi_D(w,\pi) = \frac{1}{2}w^T Hw - b^T\pi \\ \text{subject to} & Hw - A^T\pi - z = -c, \quad z \geq 0. \end{array} \tag{4.2}$$

The relationship between the primal and the dual problems is evident from the optimality conditions for (4.2) provided by the following result. The stationarity conditions for the dual are the feasibility conditions of the primal and vice versa.

Result 4.1.1 (Dual QPoptimality conditions)**.** *The point* (w^*, π^*, z^*) *is a solution to the dual QP* (4.2) *if and only if*

(a) $Hw^* - A^T\pi^* - z^* = -c$ *and* $z^* \geq 0$*;*

(b) *there exists a vector* x^* *such that* (i) $Hw^* = Hx^*$*,* (ii) $Ax^* = b$*,* (iii) $x^* \geq 0$*, and* (iv) $x^* \cdot z^* = 0$.

Second-order conditions are unnecessary because H is positive semidefinite. If the solution of the primal problem is unbounded, then the dual is infeasible. Similarly, if the dual is unbounded, then the primal is infeasible. If the dual has a bounded solution, then part (b) implies that x^*, the Lagrange multiplier vector for the dual, is a KKT point for the primal, and hence constitutes a primal solution. Moreover, if the dual has a bounded solution and H is nonsingular, then $w^* = x^*$.

Methods that solve the dual are useful because the dual formulation does not require feasibility with respect to the equality constraints $Ax = b$. For example, in branch-and-cut methods for mixed-integer nonlinear programming (MINLP), introducing a new cut constraint produces a new QP that is better solved by dual methods than primal methods. When a cut is generated, then (i) a new row and new column are added to the constraint matrix A, (ii) a zero element is added to the objective vector c, and (iii) the Hessian is extended to include a zero row and column. These changes give a new QP with data $\widehat{A}$, $\widehat{b}$, $\widehat{c}$ and $\widehat{H}$. The new column of $\widehat{A}$ corresponds to the unit vector associated with the new slack column. An obvious initial basis for the new problem is

$$\widehat{A}_B = \begin{pmatrix} A_B & 0 \\ a^T & 1 \end{pmatrix},$$

so the new basic solution $\widehat{x}_B$ is the old solution x_B augmented by the new slack, which is infeasible. The infeasible slack implies that it is necessary to go into phase 1 before solving the primal QP. However, by solving the dual QP, then we have an initial feasible subspace minimizer for the dual based on $\widehat{x}_B$ such that $\widehat{A}_B\widehat{x}_B = \widehat{b}$ and $\widehat{z} = \widehat{c} + \widehat{H}\widehat{x} - \widehat{A}^T\widehat{\pi}$. In this situation, the vector $\widehat{\pi}$ may be chosen as the old π augmented by a zero in the position of the new row of $\widehat{A}$. The new element of $\widehat{x}_B$ corresponds to the new slack, so the new elements of $\widehat{c}$ and row and column of $\widehat{H}$ are zero. This implies that $\widehat{z}$ is essentially z, and hence $\widehat{z} \geq 0$.

4.1.1 Regularized dual problem

The dual active-set method is formulated by applying the standard-form nonbinding direction method to the dual problem (4.2). The method is suitable for QPs that are not strictly convex (as in the primal case) and, as in the Bartlett-Biegler approach, the method may be implemented without the need for customized linear algebra software. However, the method cannot be applied directly to (4.2). If H is singular, then a bounded dual solution (w, π, z) is not unique because $(w + q, \pi, z)$ is also a solution for all $q \in \operatorname{null}(H)$. In addition, a working-set matrix for (4.2) has the form

$$\begin{pmatrix} H & -A^T & I \\ 0 & 0 & E^T \end{pmatrix},$$

where E is some submatrix of the identity matrix I_n. If H is singular, then the working-set matrix will be rank deficient, so that the dual has no subspace minimizers—i.e., the reduced Hessian is positive semidefinite and singular at every subspace stationary point. These difficulties may be overcome by including additional artificial equality constraints in the dual that do not alter the optimal dual objective. Let Z be a matrix whose columns form a basis for the null space of H. The regularized dual problem is defined as

$$\begin{array}{ll} \underset{w,z\in\mathbb{R}^n,\pi\in\mathbb{R}^m}{\text{minimize}} & \varphi_D(w) = \tfrac{1}{2}w^THw - b^T\pi \\ \text{subject to} & Hw - A^T\pi - z = -c, \quad Z^Tw = 0, \quad z \geq 0. \end{array} \tag{4.3}$$

The additional constraint $Z^Tw = 0$ forces w to lie in the range-space of H. The following result shows that any solution of the regularized dual (4.3) is a solution of the original dual (4.2).

Result 4.1.2 (Optimality of the regularized dual QP)**.** *A bounded solution* (w^*, π^*, z^*) *of the regularized dual QP* (4.3) *is a solution of the dual QP* (4.2)*.*

Proof. The regularized dual (4.3) is a convex problem in standard form. The optimality conditions follow from part (a) of Result 3.1.1. If (w^*, π^*, z^*) is a bounded solution of the regularized dual, then $Z^T w^* = 0$, $Hw^* - A^T\pi^* - z^* = -c$, $z^* \geq 0$, and there exist vectors x^*, y^* and q^* such that

$$\begin{pmatrix} Hw^* \\ -b \\ 0 \end{pmatrix} = \begin{pmatrix} Z & H \\ 0 & -A \\ 0 & -I \end{pmatrix} \begin{pmatrix} q^* \\ x^* \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ y^* \end{pmatrix}, \tag{4.4}$$

with $y^* \geq 0$, and $y^* \cdot z^* = 0$. The first block of equations in (4.4) gives $H(w^* - x^*) = Zq^*$, which implies that $Zq^* = 0$ because Zq^* lies in both the null space and range space of H. As the columns of Z are linearly independent, it must hold that $q^* = 0$. The second block of equations implies $Ax^* = b$, and the third implies $y^* = x^*$. Hence (w^*, π^*, z^*) satisfies $Hw^* - A^T\pi^* - z^* = -c$, and x^* is such that $Hw^* = Hx^*$, $Ax^* = b$, with $x^* \cdot z^* = 0$ and $x^* \geq 0$. It follows that (w^*, π^*, z^*) and the dual "π-vector" x^* satisfies the optimality conditions for the dual QP (4.2). $\square$

The restriction that $w \in \text{range}(H)$ implies that the optimal w is the unique vector of least two-norm that satisfies $Hw - A^T\pi - z = -c$. In many cases the null-space basis Z may be determined by inspection. For example, consider a QP with H and A of the form

$$H = \begin{pmatrix} \bar{H} & 0 \\ 0 & 0 \end{pmatrix} \quad \text{and} \quad A = \begin{pmatrix} \bar{A} & -I_m \end{pmatrix}, \tag{4.5}$$

where $\bar{H}$ is an $(n - m) \times (n - m)$ positive-definite matrix. (This format arises when a strictly convex QP with all-inequality constraints $\bar{A}x \geq \bar{b}$ is converted to standard form (see (1.3)). In this case, Z is the $(n + m) \times m$ matrix consisting of a zero $n \times m$ block and the identity I_m. Similarly if the QP is a linear program, then $Z = I_n$ and $w = 0$.

4.2 A Dual Nonbinding-Direction Method

Consider a feasible point (w, π, z, x) for the dual QP (4.3). To make the notation for the dual algorithm consistent with the notation for the primal algorithm in Chapter 3, the working (or basic) set $\mathcal{B}$ will be used to denote the n_B indices of inequality constraints in the working set for the dual QP. The associated *working-set matrix* has the form

$$A_w = \begin{pmatrix} Z^T & & \\ H & -A^T & -I \\ & & P_B^T \end{pmatrix}, \tag{4.6}$$

where P_B is the $n \times n_B$ matrix with unit columns $\{ e_i \}$ such that $i \in \mathcal{B}$. As in the primal case, the working-set matrix must have full row rank. H being singular causes no complications because the additional constraints $Z^T w = 0$ ensure that A_w will have full row rank. In the primal standard-form algorithm, independence of the rows of A_w implies independence of the columns of A_B. In the dual context, however, the independence of the columns of A_B must be imposed explicitly.

As the dual problem is convex (i.e., H is positive semidefinite), the reduced Hessian $Z_B^T H Z_B$ is always positive semidefinite, where the columns of Z_B form a basis for the null space of A_B. By Corollary 1.3.1, implies that the reduced KKT matrix K_B is nonsingular if and only if $Z_B^T H Z_B$ is positive definite. Moreover, these conditions are equivalent to K_B having inertia $(n_B, m, 0)$. Therefore, for the remainder of this section, we discuss the nonsingularity of K_B instead its inertia. In the following result, we show that the full KKT matrix of the dual problem is nonsingular if and only if the reduced KKT matrix

$$K_B = \begin{pmatrix} H_B & A_B^T \\ A_B & 0 \end{pmatrix} \tag{4.7}$$

is nonsingular.

Result 4.2.1 (Nonsingularity of the dual KKT matrix)**.** *Let $\mathcal{B}$ be a basic set with an associated working-set matrix. Then the full KKT matrix K of the dual problem* (4.3) *is nonsingular if and only if the reduced KKT matrix K_B is nonsingular.*

Proof. Let K_B denote the reduced KKT matrix in (4.7) and assume that K_B is nonsingular. It suffices to show that K has the trivial null space, i.e., if $Ku = 0$, then $u = 0$. Let K and u be partitioned conformably as

$$K = \left(\begin{array}{ccc|ccc} H & 0 & 0 & Z & H & 0 \\ 0 & 0 & 0 & 0 & -A & 0 \\ 0 & 0 & 0 & 0 & -I & P_B \\ \hline Z^T & 0 & 0 & 0 & 0 & 0 \\ H & -A^T & -I & 0 & 0 & 0 \\ 0 & 0 & P_B^T & 0 & 0 & 0 \end{array}\right), \quad \text{and} \quad u = \left(\begin{array}{c} u_1 \\ u_2 \\ u_3 \\ \hline u_4 \\ u_5 \\ u_6 \end{array}\right). \tag{4.8}$$

The first block of the system $Ku = 0$ yields $u_4 = 0$ and $Hu_1 = -Hu_5$. Furthermore, the third and sixth blocks imply that $(u_3)_B = 0$ and $(u_5)_N = 0$. Combining these identities with the fifth block, $Hu_5 + A^T u_2 + u_3 = 0$ and partitioning the resulting vectors into their basic and nonbasic components, gives

$$\begin{pmatrix} H_B & A_B^T \\ A_B & 0 \end{pmatrix} \begin{pmatrix} (u_5)_B \\ u_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

with the second block of this system coming from the second block of (4.8). Since the reduced KKT system is nonsingular by assumption, we have $u_2 = 0$ and $u_5 = 0$. Moreover, since u_3 is a linear combination of u_2 and u_5, it holds that $u_3 = 0$. The third block further implies $u_6 = 0$.

Because $u_5 = 0$, then $Hu_1 = 0$ and $Z^T u_1 = 0$ from the first and fourth blocks of (4.8). Then, u_1 lies in both the range space and null space, and $u_1 = 0$. Therefore, $Ku = 0$ implies $u = 0$ and K is nonsingular.

Now assume that K is nonsingular, and that there exist vectors x_B and y such that

$$\begin{pmatrix} H_B & A_B^T \\ A_B & 0 \end{pmatrix} \begin{pmatrix} x_B \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

If x is defined such that $x = P_B^T x_B$, then $Ax = A_B x_B = 0$. Also, let u_1 be the range-space portion of x, in which case $Hu_1 = Hx$ and $Z^T u_1 = 0$. Also, define $u_3 = -Hx - A^T y$.

Then,

$$\left(\begin{array}{ccc|ccc} H & 0 & 0 & Z & H & 0 \\ 0 & 0 & 0 & 0 & -A & 0 \\ 0 & 0 & 0 & 0 & -I & P_B \\ \hline Z^T & 0 & 0 & 0 & 0 & 0 \\ H & -A^T & -I & 0 & 0 & 0 \\ 0 & 0 & P_B^T & 0 & 0 & 0 \end{array}\right) \left(\begin{array}{c} u_1 \\ y \\ u_3 \\ \hline 0 \\ x \\ x_B \end{array}\right) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

As K is nonsingular by assumption, we must have that $x_B = 0$ and $y = 0$. It follows that K_B must also be nonsingular. □

The properties of a dual subspace stationary point and a dual second-order-consistent basis are summarized in the following result.

Result 4.2.2 (Dual stationary point and dual subspace minimizer)**.** *Let* (w, π, z, x) *be a dual-feasible point with basic set* $\mathcal{B}$*.*

(a) *If* (w, π, z, x) *is a* dual stationary point with respect to $\mathcal{B}$*, then* $Hw = Hx$ *and* $Ax = b$ *with* $x_N = 0$.

(b) *Furthermore, if* $\mathcal{B}$ *is a* dual second-order-consistent basis *for the dual problem* (4.3)*, then the reduced KKT matrix*

$$K_B = \begin{pmatrix} H_B & A_B^T \\ A_B & 0 \end{pmatrix}$$

is nonsingular.

Proof. For (w, π, z, x) to be a stationary point, the gradient of the objective at this point must lie in the range space of the transpose of the working-set matrix (4.6). Thus, at a stationary point, there must exist vectors q, x and y_B such that

$$\nabla \varphi_D(w, \pi, z) = \begin{pmatrix} Hw \\ -b \\ 0 \end{pmatrix} = \begin{pmatrix} Z & H & 0 \\ 0 & -A & 0 \\ 0 & -I & P_B \end{pmatrix} \begin{pmatrix} q \\ x \\ y_B \end{pmatrix}.$$

As in the proof of Result 4.1.2, $q = 0$, so that $Hw = Hx$ and $Ax = b$. The last block of the system implies $x = P_B y_B$ so that $x_N = 0$.

For $\mathcal{B}$ to be a second-order-consistent basis, the full KKT matrix of the dual must be nonsingular (with restrictions on the sign of its eigenvalues being unnecessary, as explained above), which implies that K_B is also nonsingular by Result 4.2.1. $\square$

At a subspace stationary point, the variables x (the dual variables of the dual problem) define a basic solution of the primal equality constraints. Moreover, the dual equality constraints imply that $z = Hw - A^T\pi + c = g(w) - A^T\pi = g(x) - A^T\pi$, which are the primal reduced-costs corresponding to both w and x. With the regularizing constraints $Z^Tw = 0$, the vectors w and x differ by a vector in the null space of H. It will be shown below that if the QP gradient $g(w) = c + Hw$ is known, the vector w need not be computed explicitly.

Let (w, π, z) be a nonoptimal dual subspace minimizer. Since the point is not optimal, there is at least one negative component of the dual multiplier vector x, say $x_{\beta_r} < 0$. The application of the nonbinding-direction method of Chapter 3 to the dual gives a search direction $(\Delta w, \Delta\pi, \Delta z)$ that is feasible for the dual working-set constraints, and increases a designated constraint with a negative multiplier. The direction satisfies

$$Z^T\Delta w = 0, \quad H\Delta w - A^T\Delta\pi - \Delta z = 0, \text{ and } P_B^T\Delta z = e_r.$$

These equations are incorporated into the following system of equations that are the dual-algorithm equivalent to System 1 (3.3):

$$\left(\begin{array}{ccc|ccc} H & 0 & 0 & Z & H & 0 \\ 0 & 0 & 0 & 0 & -A & 0 \\ 0 & 0 & 0 & 0 & -I & P_B \\ \hline Z^T & 0 & 0 & 0 & 0 & 0 \\ H & -A^T & -I & 0 & 0 & 0 \\ 0 & 0 & P_B^T & 0 & 0 & 0 \end{array}\right) \left(\begin{array}{c} \Delta w \\ \Delta\pi \\ \Delta z \\ \hline -\Delta q \\ -\Delta x \\ -\Delta y_B \end{array}\right) = \left(\begin{array}{c} 0 \\ 0 \\ 0 \\ \hline 0 \\ 0 \\ e_r \end{array}\right). \tag{4.9}$$

The first block implies that $H\Delta w = Z\Delta q + H\Delta x$, so that $Z\Delta q$ lies in the range space and null space of H. It follows that $Z\Delta q = 0$ and $\Delta q = 0$. Therefore, $H\Delta w = H\Delta x$. In addition, the third block $\Delta x = P_B\Delta y_B$ implies that $\Delta x_N = 0$ and $\Delta x_B = \Delta y_B$.

As in the primal standard-form case, the search direction may be computed from the smaller system

$$\begin{pmatrix} H_B & A_B^T \\ A_B & 0 \end{pmatrix} \begin{pmatrix} \Delta x_B \\ -\Delta\pi \end{pmatrix} = \begin{pmatrix} e_r \\ 0 \end{pmatrix}, \tag{4.10}$$

with $\Delta x_N = 0$, $\Delta z_B = e_r$, $\Delta z_N = (H\Delta x - A^T\Delta\pi)_N$ and $H\Delta w = H\Delta x$.

Result 4.2.3 (Properties of a dual nonbinding search directions)**.** *Let Δw, $\Delta\pi$, Δz and Δx satisfy the KKT system* (4.9)*. Then the following properties are satisfied*

(a) $\Delta x^T H \Delta x = [\Delta x_B]_r$;

(b) $\big(\Delta w, \Delta\pi, \Delta z\big)^T \nabla\varphi_D(w, \pi, z) = [x_B]_r$.

Proof. For part (a), since $A\Delta x = 0$, we have

$$\begin{aligned}\Delta x^T H \Delta x &= \Delta x^T(H\Delta x - A^T\Delta\pi)\\ &= \Delta x^T \Delta z = \Delta x_B^T \Delta z_B = \Delta x_B^T e_r = [\Delta x_B]_r.\end{aligned}$$

For part (b), we use the definition of the gradient of the dual problem to give

$$\begin{aligned}\big(\Delta w, \Delta\pi, \Delta z\big)^T \nabla\varphi_D &= \big(\Delta w, \Delta\pi, \Delta z\big)^T \begin{pmatrix} Hw \\ -b \\ 0 \end{pmatrix}\\ &= \Delta w^T H w - b^T \Delta\pi\\ &= \Delta w^T H x - x^T A^T \Delta\pi \quad \text{(because } Ax = b \text{ and } Hx = Hw)\\ &= x^T(H\Delta w - A^T\Delta\pi)\\ &= x^T \Delta z = x_B^T e_r \qquad\qquad \text{(because } x_N = 0)\\ &= [x_B]_r,\end{aligned}$$

as required. □

If the curvature $\Delta x^T H \Delta x$ is nonzero, the optimal step $\alpha_* = -[x_B]_r/[\Delta x_B]_r$ minimizes the dual objective $\varphi_D(w + \alpha\Delta w, \pi + \alpha\Delta\pi, z + \alpha\Delta z)$ with respect to α, and the r-th element of $x_B + \alpha_*\Delta x_B$ is zero. If x_B are interpreted as estimates of the primal variables (i.e., variables of the primal QP), then the step from x_B to $x_B + \alpha_*\Delta x_B$ increases the negative (and hence infeasible) primal variable $[x_B]_r$ until it reaches its bound of zero. If the step $\alpha = \alpha_*$ gives a feasible point for the dual inequalities (i.e., $z + \alpha_*\Delta z \geq 0$), then the next iterate is $(w + \alpha\Delta w, \pi + \alpha\Delta\pi, z + \alpha\Delta z)$. Updates to the basic set in this case are given in the following result.

Result 4.2.4 (Constraint deletion)**.** *Let (w, π, z, x) be a subspace minimizer with respect to $\mathcal{B}$. Assume that $x_{\beta_r} < 0$, and let $(\bar{w}, \bar{\pi}, \bar{z}, \bar{x}) = (w + \alpha\Delta w, \pi + \alpha\Delta\pi, z + \alpha\Delta z, x + \alpha\Delta x)$, where $(\Delta w, \Delta\pi, \Delta z, \Delta x)$ are defined by* (4.10)*, and $\alpha = \alpha_*$ is bounded. Then $(\bar{w}, \bar{\pi}, \bar{z}, \bar{x})$ is a subspace minimizer with respect to $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\}$.*

Proof. By (4.10), $A\Delta x = 0$, so that $A(x + \alpha\Delta x) = b$. Since $H\Delta w = H\Delta x$, we have $H(w + \alpha\Delta w) = H(x + \alpha\Delta x)$. The definition of $\alpha = \alpha_*$ implies that $[x + \alpha\Delta x]_{\beta_r} = 0$, so that $\bar{x}_{\bar{N}} = 0$, where $\bar{\mathcal{N}} = \mathcal{N} + \{\beta_r\}$.

Now we show that $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\}$ is a second-order-consistent basis by showing that $K_{\bar{B}}$ is nonsingular. Consider the matrix

$$M \triangleq \begin{pmatrix} K_B & e_r \\ e_r^T & \end{pmatrix}.$$

By assumption, (w, π, z) is a subspace minimizer and K_B is nonsingular, so that the Schur complement of K_B in M exists, with

$$\begin{aligned} M/K_B = -e_r^T K_B^{-1} e_r = -e_r^T \begin{pmatrix} \Delta x_B \\ -\Delta\pi \end{pmatrix} & \quad \text{(from (4.10))} \\ = -[\Delta x_B]_r \neq 0 & \quad \text{(because } \alpha_* \text{ is bounded).} \end{aligned}$$

Then $\mathrm{In}(M) = \mathrm{In}(M/K_B) + \mathrm{In}(K_B) = \mathrm{In}(-[\Delta x_B]_r) + \mathrm{In}(K_B)$, and M is nonsingular because both M/K_B and K_B are nonsingular.

Since $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\}$, a permutation can be applied to K_B such that

$$K_B = \begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix} \sim \left(\begin{array}{cc|c} H_{\bar{B}} & (h_{\beta_r})_{\bar{B}} & A_{\bar{B}}^T \\ (h_{\beta_r})_{\bar{B}}^T & h_{\beta_r,\beta_r} & a_{\beta_r}^T \\ \hline A_{\bar{B}} & a_{\beta_r} & 0 \end{array}\right).$$

Similarly, applying symmetric permutations to M gives

$$\begin{aligned} M \triangleq \begin{pmatrix} K_B & e_r \\ e_r^T & \end{pmatrix} &\sim \left(\begin{array}{ccc|c} H_{\bar{B}} & (h_{\beta_r})_{\bar{B}} & A_{\bar{B}}^T & 0 \\ (h_{\beta_r})_{\bar{B}}^T & h_{\beta_r,\beta_r} & a_{\beta_r}^T & 1 \\ A_{\bar{B}} & a_{\beta_r} & 0 & 0 \\ \hline 0 & 1 & 0 & 0 \end{array}\right) \\ &\sim \left(\begin{array}{cc|cc} h_{\beta_r,\beta_r} & 1 & (h_{\beta_r})_{\bar{B}}^T & a_{\beta_r}^T \\ 1 & 0 & 0 & 0 \\ \hline (h_{\beta_r})_{\bar{B}} & 0 & H_{\bar{B}} & A_{\bar{B}}^T \\ a_{\beta_r} & 0 & A_{\bar{B}} & 0 \end{array}\right) \triangleq \widetilde{M}. \end{aligned}$$

The leading 2×2 block of $\widetilde{M}$, denoted by E, has $\det(E) = -1$ so the Schur complement of E in $\widetilde{M}$ is

$$\begin{aligned} \widetilde{M}/E &= K_{\bar{B}} - \begin{pmatrix} (h_{\beta_r})_{\bar{B}} & 0 \\ a_{\beta_r} & 0 \end{pmatrix} \begin{pmatrix} h_{\beta_r,\beta_r} & 1 \\ 1 & 0 \end{pmatrix}^{-1} \begin{pmatrix} (h_{\beta_r})_{\bar{B}}^T & a_{\beta_r}^T \\ 0 & 0 \end{pmatrix} \\ &= K_{\bar{B}} - \begin{pmatrix} (h_{\beta_r})_{\bar{B}} & 0 \\ a_{\beta_r} & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -h_{\beta_r,\beta_r} \end{pmatrix} \begin{pmatrix} (h_{\beta_r})_{\bar{B}}^T & a_{\beta_r}^T \\ 0 & 0 \end{pmatrix} \\ &= K_{\bar{B}}, \end{aligned}$$

which implies that $\mathrm{In}(M) = \mathrm{In}(\widetilde{M}) = \mathrm{In}(\widetilde{M}/E) + \mathrm{In}(E) = \mathrm{In}(K_{\bar{B}}) + \mathrm{In}(E)$.

Thus, $\mathrm{In}(K_{\bar{B}}) = \mathrm{In}(M) - \mathrm{In}(E)$, which implies $K_{\bar{B}}$ is nonsingular since M and E are nonsingular. It follows that $\bar{\mathcal{B}}$ is a second-order-consistent basis, and $(\bar{w}, \bar{\pi}, \bar{z}, \bar{x})$ is a subspace minimizer with respect to $\bar{\mathcal{B}}$. $\square$

If α_* is unbounded, or steps to an infeasible point, then α is defined as the largest step such that z remains nonnegative, i.e., $\alpha_F = \min_{1\le i\le n_N}\{\gamma_i\}$, where

$$\gamma_i = \begin{cases} \dfrac{[z_N]_i}{-[\Delta z_N]_i} & \text{if } [\Delta z_N]_i < 0, \\ +\infty & \text{otherwise.} \end{cases}$$

If $\alpha_F < \alpha_*$, then at least one of the dual residuals is zero at $(w+\alpha\Delta w, \pi+\alpha\Delta\pi, z+\alpha\Delta z, x+\alpha\Delta x)$, and the index of one of these, say ν_s is moved to $\mathcal{B}$.

The removal of β_r from $\mathcal{B}$ is determined by a constraint dependency test that is based on the solution of a system that is analogous to System 2 of the mixed-constraint and standard-form algorithms. Let u, u_π, u_z, q, v, and u_B be the solution to the full KKT system

$$\left(\begin{array}{ccc|ccc} H & 0 & 0 & Z & H & 0 \\ 0 & 0 & 0 & 0 & -A & 0 \\ 0 & 0 & 0 & 0 & -I & P_B \\ \hline Z^T & 0 & 0 & 0 & 0 & 0 \\ H & -A^T & -I & 0 & 0 & 0 \\ 0 & 0 & P_B^T & 0 & 0 & 0 \end{array}\right) \begin{pmatrix} u \\ u_\pi \\ u_z \\ -q \\ -v \\ -u_B \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ e_{\nu_s} \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{4.11}$$

Using Result 2.2.6, linear dependence occurs if and only if the vectors u, u_π and u_z are all zero. However, it can be shown that it is unnecessary to solve the full KKT system or check all three vectors in the dependency test.

Result 4.2.5 (Test for dual constraint dependency). *Let u, u_π, u_z, q, v, and u_B be the solution to the full KKT system* (4.11). *Assume that (w, π, z) is a subspace minimizer for the dual. Assume that the ν_s-th dual inequality constraint is blocking at $(\bar w, \bar\pi, \bar z) = (w, \pi, z)+\alpha(\Delta w, \Delta\pi, \Delta z)$, where $(\Delta w, \Delta\pi, \Delta z)$ satisfies* (4.10). *Then*

(a) $u = 0$ *if and only if* $u_\pi = 0$, *and* $u_z = 0$ *if and only if* $Hv = 0$, *where* $v = P_B u_B + e_{\nu_s}$;

(b) *the vectors u_B and u_π satisfy the reduced KKT system*

$$\begin{pmatrix} H_B & A_B^T \\ A_B & 0 \end{pmatrix} \begin{pmatrix} u_B \\ -u_\pi \end{pmatrix} = -\begin{pmatrix} (h_{\nu_s})_B \\ a_{\nu_s} \end{pmatrix}, \tag{4.12}$$

with $v = P_B u_B + e_{\nu_S}$ *and* $u_z = Hv - A^T u_\pi$;

(c) *the gradient of the ν_s-th dual constraint e_{ν_s} is linearly independent of the gradients of the working-set constraints if and only if* $Hv = H(P_B u_B) + h_{\nu_s} \neq 0$;

(d) $[u_B]_r = -[\Delta z]_{\nu_s} > 0$; *and if* $u \neq 0$, *then* $[u_z]_{\nu_s} > 0$.

Proof. The last block of (4.11) implies $(u_z)_B = 0$. Notice that if $u = 0$, then $u_z = -A^T u_\pi$, so that $0 = -A_B^T u_\pi$. Since A_B^T has linearly independent columns, $u_\pi = 0$ and $u_z = 0$. If $u_\pi = 0$ and $u_z = 0$, then $Hu = 0$ and $Z^T u = 0$ and $u = 0$. Thus, $u = 0$ if and only if $u_\pi = u_z = 0$.

If $Hv = 0$, then $Hu = 0$ and the fourth block of 4.11 implies $Z^T u = 0$, so that u is in both the null space and range space of H. Thus $u = 0$. If $u = 0$, then $Hu = Hv = 0$.

For part (b), the first block implies $q = 0$, and $Hu = Hv$ and $Z^T u = 0$. The third block gives $v = P_B u_B + e_{\nu_s}$. Combining these results implies $Hu = H(P_B u_B) + h_{\nu_s}$ and $Av = A_B u_B + a_{\nu_s} = 0$. By the fifth block, $u_z = Hu - A^T u_\pi = H(P_B u_B) + h_{\nu_s}$. Since the last block of the system implies $(u_z)_B = 0$, we have that

$$H_B u_B - A_B^T u_\pi = -(h_{\nu_s})_B \text{ and } A_B u_B = -a_{\nu_s},$$

so that u_B and u_π satisfy the reduced KKT system

$$\begin{pmatrix} H_B & A_B^T \\ A_B & 0 \end{pmatrix} \begin{pmatrix} u_B \\ -u_\pi \end{pmatrix} = - \begin{pmatrix} (h_{\nu_s})_B \\ a_{\nu_s} \end{pmatrix}.$$

Hence part (c) follows by part (a) and Result 2.2.6.

For part (d),

$$\begin{aligned} [u_B]_r = e_r^T u_B &= \Delta z_B^T u_B = u_B^T (H_B \Delta w_B - A_B^T \Delta\pi) \\ &= -(h_{\nu_s})_B^T \Delta w_B + a_{\nu_s}^T \Delta\pi \\ &= -(\Delta w^T H e_{\nu_s} - \Delta\pi^T A e_{\nu_s}) = -e_{\nu_s}^T (H\Delta w - A^T \Delta\pi) \\ &= -e_{\nu_s}^T \Delta z = -[\Delta z]_{\nu_s} > 0, \end{aligned}$$

where the last inequality holds since $z_{\nu_s} \geq 0$ is a blocking constraint and $[\Delta z]_{\nu_s} < 0$.

Using the fact that $e_{\nu_s} = v - P_B u_B$ implies

$$\begin{aligned} u_z^T e_{\nu_s} = u_z^T (Hv - P_B u_B) &= v^T H u_z \text{ since } P_B^T u_z = 0 \\ &= v^T (Hv - A^T u_\pi) = v^T H v > 0. \end{aligned}$$

The last inequality follows since $Av = 0$ and H is positive definite in the null space of A at a subspace minimizer. □

Once constraint dependency is determined, the basic set for the next iterate is updated according to the following result.

Result 4.2.6 (Basic set updates)**.** *Let (w, π, z, x) be a subspace minimizer with respect to $\mathcal{B}$. Assume that the ν_s-th dual constraint is blocking at $(\bar{w}, \bar{\pi}, \bar{z}, \bar{x}) = (w, \pi, z, x) + \alpha(\Delta w, \Delta\pi, \Delta z, \Delta x)$, where the search directions satisfy* (4.10)*. Let u_B, u_π and v be defined by* (4.12)*.*

(a) *If the ν_s-th constraint gradient is linearly independent of the working-set constraint gradients* (4.6)*, then $(\bar{w}, \bar{\pi}, \bar{z}, \bar{x})$ is a subspace minimizer with respect to $\bar{\mathcal{B}} = \mathcal{B} + \{\nu_s\}$.*

(b) *If the ν_s-th constraint gradient is linearly dependent on the working-set constraint gradients* (4.6)*, then the scalar $\sigma = -[x + \alpha\Delta x]_{\beta_r}/[u_B]_r$ is well defined. Moreover, $(\bar{w}, \bar{\pi}, \bar{z}, \bar{x})$ is a subspace minimizer with respect to $\bar{\mathcal{B}} = \mathcal{B} + \{\nu_s\} - \{\beta_r\}$, and the associated multipliers $\bar{x}$ are given by $x + \alpha\Delta x + \sigma v$.*

Proof. Assume the constraint gradients are linearly independent. Stationarity holds trivially since $x_N = 0$ at a stationary point and (4.10) implies $\Delta x_N = 0$.

Now let K_B and $K_{\bar{B}}$ denote the matrices associated with basic sets $\mathcal{B}$ and $\bar{\mathcal{B}}$. We must show that $K_{\bar{B}}$ is nonsingular.

Define $\widetilde{K}_{\bar{B}}$ as the permuted version of $K_{\bar{B}}$ such that

$$\widetilde{K}_{\bar{B}} = Q^T K_{\bar{B}} Q = \left(\begin{array}{cc|c} H_B & A_B^T & (h_{\nu_s})_{\mathcal{B}} \\ A_B & & a_{\nu_s} \\ \hline (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T & h_{\nu_s,\nu_s} \end{array}\right),$$

where Q is a permutation matrix. By assumption, the matrix K_B is nonsingular, so the Schur complement of K_B in $\widetilde{K}_{\bar{B}}$ exists. Using Result 1.3.3, the matrix $\widetilde{K}_{\bar{B}}$ is nonsingular if and only if $\widetilde{K}_{\bar{B}}/K_B$ is nonsingular. We can see that

$$\begin{aligned}
\widetilde{K}_{\bar{B}}/K_B &= h_{\nu_s,\nu_s} - \begin{pmatrix} (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T \end{pmatrix} K_B^{-1} \begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} \\ a_{\nu_s} \end{pmatrix} \\
&= h_{\nu_s,\nu_s} + \begin{pmatrix} (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T \end{pmatrix} \begin{pmatrix} u_B \\ -u_\pi \end{pmatrix} \qquad \text{(from (4.12))} \\
&= e_{\nu_s}^T H e_{\nu_s} + (h_{\nu_s})_B^T u_B - e_{\nu_s}^T A^T u_\pi \\
&= e_{\nu_s}^T \big(H(P_B u_B) + h_{\nu_s} - A^T u_\pi\big) = e_{\nu_s}^T u_z.
\end{aligned}$$

Result 4.2.5 implies that $[u_z]_{\nu_s} > 0$. Thus $K_{\bar{B}}$ is nonsingular with respect to $\bar{\mathcal{B}}$ and the next iterates remains a subspace minimizer.

For part (b), we begin by observing that $Hv = 0$ and $u_\pi = u_z = 0$. Let $\bar{\mathcal{B}} = \mathcal{B} + \{\nu_s\} - \{\beta_r\}$. By definition, $v = P_B u_B + e_{\nu_s}$, so that $v_B = u_B$. Because of the definition of σ, it must hold that $[x + \alpha \Delta x + \sigma v]_{\beta_r} = 0$. Then the next iterate is a stationary point with respect to $\bar{\mathcal{B}}$. It remains to show that $K_{\bar{B}}$ is nonsingular.

Let y denote the vector $(u_B, 0)$. Then since $u_\pi = 0$, (4.12) implies

$$K_B y = -\begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} \\ a_{\nu_s} \end{pmatrix}.$$

The updated condensed KKT matrix can be written in terms of the symmetric rank-one modification to K_B:

$$\begin{aligned}
K_{\bar{B}} &= K_B + (K_B y - K_B e_r)e_r^T + e_r(K_B y - K_B e_r)^T + e_r\big((y - e_r)^T K_B (y - e_r)\big)e_r^T \\
&= \big(I + e_r(y - e_r)^T\big) K_B \big(I + (y - e_r)e_r^T\big).
\end{aligned}$$

Since $[u_B]_r \neq 0$ by part (d) of Result 4.2.5, the matrix $I + e_r(y - e_r)^T$ and its transpose are nonsingular. Therefore, $K_{\bar{B}}$ is nonsingular if and only if K_B is nonsingular. $\square$

Algorithm 4.1 summarizes the nonbinding-direction method for solving the dual of a convex quadratic programming problem in standard form.

Algorithm 4.1: Dual nonbinding-direction method for a convex QP in standard form

Find x_0 such that $Ax_0 = b$.

$[x, \pi, \mathcal{B}, \mathcal{N}] = \texttt{subspaceMin}(x_0)$;

$g = c + Hx; \quad z = c + Hx - A^T\pi$;

$\beta_r = \operatorname{argmin}_i\{[x_B]_i\}$;

while $x_{\beta_r} < 0$ **do**

  Solve $\begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix} \begin{pmatrix} \Delta x_B \\ -\Delta\pi \end{pmatrix} = \begin{pmatrix} e_r \\ 0 \end{pmatrix}; \quad \Delta z = H\Delta x - A^T\Delta\pi$;

  $\alpha_F = \texttt{minRatioTest}(z_N, \Delta z_N)$;

  if $[\Delta x_B]_r > 0$ **then** $\alpha_* = -[x_B]_r/[\Delta x_B]_r$ **else** $\alpha_* = +\infty$;

  $\alpha = \min\{\alpha_*, \alpha_F\}$;

  if $\alpha = +\infty$ **then stop**; [the primal is infeasible]

  $x \leftarrow x + \alpha\Delta x; \quad g \leftarrow g + \alpha H\Delta x$;

  $\pi \leftarrow \pi + \alpha\Delta\pi; \quad z \leftarrow z + \alpha\Delta z$;

  if $\alpha_F < \alpha_*$ **then** [add the dual working-set constraint ν_s]

    Find the blocking constraint index ν_s;

    Solve $\begin{pmatrix} H_B & A_B^T \\ A_B & 0 \end{pmatrix} \begin{pmatrix} u_B \\ -u_\pi \end{pmatrix} = -\begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} \\ a_{\nu_s} \end{pmatrix}, \quad v = P_B u_B + e_{\nu_s}$;

    if $Hv = 0$ **then** $\sigma = -[x_B]_r/[u_B]_r$ **else** $\sigma = 0$;

    $\mathcal{B} \leftarrow \mathcal{B} + \{\nu_s\}; \quad \mathcal{N} \leftarrow \mathcal{N} - \{\nu_s\}$;

    $x \leftarrow x + \sigma v$;

    $g \leftarrow g + \sigma Hv; \quad z \leftarrow g - A^T\pi$;

  end;

  if $x_{\beta_r} = 0$ **then** [delete the dual working-set constraint β_r]

    $\mathcal{B} \leftarrow \mathcal{B} - \{\beta_r\}; \quad \mathcal{N} \leftarrow \mathcal{N} + \{\beta_r\}$;

    $\beta_r = \operatorname{argmin}_i\{[x_B]_i\}$;

  end;

  $k \leftarrow k + 1$;

end do

The definition of the updates to the search directions for the linearly independent constraint case are summarized in the following result.

Result 4.2.7 (Direction updates)**.** *Assume that* (x,π,z) *is a subspace minimizer with respect to* $\mathcal{B}$*, and that equations* (4.10) *and* (4.12) *hold. Then if the gradient of the blocking bound* $z_{\nu_s} \ge 0$ *at* $x+\alpha\Delta x$ *is linearly independent of the working-set constraints* (4.6) *defined by* $\mathcal{B}$*, then the vectors* $\Delta x_B + \rho u_B$ *and* $\Delta\pi + \rho u_\pi$ *such that* $\rho = -[\Delta z]_{\nu_s}/[u_z]_{\nu_s}$ *are well-defined, and satisfy*

$$K_{\bar{B}} \begin{pmatrix} \Delta x_B + \rho u_B \\ \rho \\ -(\Delta\pi + \rho u_\pi) \end{pmatrix} = \begin{pmatrix} e_r \\ 0 \\ 0 \end{pmatrix},$$

which is the KKT equation (4.10) *for the basic set* $\bar{\mathcal{B}} = \mathcal{B} + \{\nu_s\}$.

Proof. Since the blocking constraint is linearly independent of the basic-set constraints, $[u_z]_{\nu_s} \ne 0$ by part (d) of Result 4.2.5, so that ρ is well-defined.

Let $K_{\bar{B}}$ be a permuted version of the KKT matrix for $\bar{\mathcal{B}}$ such that

$$K_{\bar{B}} = \left(\begin{array}{cc|c} H_B & A_B^T & (h_{\nu_s})_{\mathcal{B}} \\ A_B & 0 & a_{\nu_s} \\ \hline (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T & h_{\nu_s\nu_s} \end{array}\right).$$

Then the following equations hold:

$$\left(\begin{array}{cc|c} H_B & A_B^T & (h_{\nu_s})_{\mathcal{B}} \\ A_B & 0 & a_{\nu_s} \\ \hline (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T & h_{\nu_s\nu_s} \end{array}\right) \left(\begin{array}{c} \Delta x_B \\ -\Delta\pi \\ \hline 0 \end{array}\right) = \left(\begin{array}{c} e_r \\ 0 \\ \hline (h_{\nu_s})_{\mathcal{B}}^T \Delta x_B - a_{\nu_s}^T \Delta\pi \end{array}\right) \tag{4.13}$$

and

$$\left(\begin{array}{cc|c} H_B & A_B^T & (h_{\nu_s})_{\mathcal{B}} \\ A_B & 0 & a_{\nu_s} \\ \hline (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T & h_{\nu_s\nu_s} \end{array}\right) \left(\begin{array}{c} \rho u_B \\ -\rho u_\pi \\ \hline \rho \end{array}\right) = \rho \left(\begin{array}{c} 0 \\ 0 \\ h_{\nu_s,\nu_s} + (h_{\nu_s})_{\mathcal{B}}^T u_B - a_{\nu_s}^T u_\pi \end{array}\right). \tag{4.14}$$

If ρ is defined as $\rho = -[\Delta z]_{\nu_s}/[u_z]_{\nu_s}$, then notice that

$$[\Delta z]_{\nu_s} = e_{\nu_s}^T (H\Delta x - A^T\Delta\pi) = (h_{\nu_s})_{\mathcal{B}}^T \Delta x_B - a_{\nu_s}^T \Delta\pi,$$

and

$$[u_z]_{\nu_s} = e_{\nu_s}^T (Hv - A^T u_\pi) = h_{\nu_s,\nu_s} + e_{\nu_s}^T H(P_B u_B) - a_{\nu_s}^T u_\pi = h_{\nu_s,\nu_s} + (h_{\nu_s})_{\mathcal{B}}^T u_B - a_{\nu_s}^T u_\pi,$$

which are the expressions in the right-hand-sides of (4.13) and (4.14). Summing the two equations yields

$$\left(\begin{array}{cc|c} H_B & A_B^T & (h_{\nu_s})_{\mathcal{B}} \\ A_B & 0 & a_{\nu_s} \\ \hline (h_{\nu_s})_{\mathcal{B}}^T & a_{\nu_s}^T & h_{\nu_s\nu_s} \end{array}\right) \left(\begin{array}{c} (\Delta x_B + \rho u_B) \\ -(\Delta\pi + \rho u_\pi) \\ \hline \rho \end{array}\right) = \begin{pmatrix} e_r \\ 0 \\ 0 \end{pmatrix},$$

which is System 1 (4.10) for $\bar{\mathcal{B}}$. $\square$

4.2.1 Dual linear programming

If H is zero, then the primal QP is a linear program. In this case we may choose Z as the identity matrix for the regularized problem (4.3). It follows from Result 4.2.2 that (w, π, z) is a subspace minimizer if A_B is nonsingular—i.e., it is square with rank m. In this case, equations (4.10) and (4.12) give

$$-A_B^T \Delta\pi = e_r, \quad A_B \Delta x_B = 0, \quad A_B^T u_\pi = 0, \text{ and } A_B u_B = -a_{\nu_s},$$

with $u_z = -A^T u_\pi$. A_B being nonsingular implies $\Delta x_B = 0$ so $\Delta x = 0$ and $u_\pi = 0$, so that $u_z = 0$. By part (a) of Result 4.2.5, $Hv = 0$, so that the linearly dependent case always applies and the index β_r is replaced by ν_s in $\mathcal{B}$, as in the dual simplex method. The update for the dual multiplier x defined by part (b) of Result 4.2.6 is given by $\bar{x} = x + \sigma v$, where $\sigma = -[x_B]_r/[u_B]_r$, and $v = P_B u_B + e_{\nu_s}$.

4.2.2 Degeneracy of the dual QP

Suppose that (w, π, z) is a feasible point for the regularized dual QP (4.3) such that r of the z-variables are at their bounds. If (w, π, z) is degenerate for the dual constraints, it must hold that r must be greater than the difference between the number of variables and equality constraints. It follows that if (w, π, z) is degenerate, then

$$r > (n + n + m) - (n + n_z) = n + m - n_z = \text{rank}(H) + m,$$

where n_z is the number of columns in the null-space basis Z. If H is nonsingular, then $Z = 0$ and a degenerate (w, π, z) would require more than $n + m$ of the n z-variables to be on their bounds, which is clearly impossible. It follows that if the primal QP is strictly convex, then there are no degenerate points for the dual.

In the general case, if $m + \text{rank}(H) \geq n$ for the dual (4.2), then there are no degenerate points. In this situation, Algorithm 4.1 cannot cycle, and will either terminate with an optimal solution or declare the dual problem to be unbounded. Observe that this nondegeneracy property does not hold for a dual linear program, but it does hold for strictly convex problems, and for any QP with H and A given by (4.5).

5 Finding an Initial Point

Thus far, discussions have been focused on the optimality phase of the active-set method. In this chapter, methods for finding the initial point for our algorithms are discussed. Section 5.1 reviews phase 1 methods for finding a feasible point such that $Ax = b$ and $x \geq 0$. Then, the process of moving to a stationary point is explained in Section 5.3. Lastly, Section 5.2 describes methods for finding a second-order-consistent basis.

5.1 Getting Feasible

The process of finding a feasible point for the constraints $Ax = b$ and $x \geq 0$ during phase 1 of the active-set methods is described in this section. There are generally two approaches. The first, common in linear programming, is to find an x that satisfies $Ax = b$, and then iterate (if necessary) to satisfy the bounds $x \geq 0$. The second method defines a nonnegative x and then iterates to satisfy $Ax = b$. We use the former approach and assume that the initial iterate x_0 satisfies $Ax = b$ (such an x_0 must exist because A has full row rank by assumption).

Suppose that the bounds $x \geq 0$ are written in the equivalent form $x = u - v$, $u \geq 0$ and $v = 0$. The idea is to relax the equality constraint $v = 0$ by minimizing some norm of v. Choosing the one-norm gives the following piecewise-linear program for a feasible point:

$$\underset{x,u,v\in\mathbb{R}^n}{\text{minimize}} \ \|v\|_1 \quad \text{subject to} \quad Ax = b, \quad x = u - v, \quad u \geq 0.$$

By adding the restriction that $v \geq 0$, the one-norm objective may be replaced by $e^T v$, giving the conventional linear program

$$\underset{x,u,v\in\mathbb{R}^n}{\text{minimize}} \ e^T v \quad \text{subject to} \quad Ax = b, \quad x = u - v, \quad u \geq 0, \quad v \geq 0. \tag{5.1}$$

The vectors u and v are referred to as *elastic variables.* At the optimal solution, u and v are the magnitudes of the positive and negative parts of the vector x that is closest in one-norm to the positive orthant and satisfies $Ax = b$. If the constraints are feasible, then $v = 0$ and $x\,(= u) \geq 0$.

At an initial x_0 satisfying $Ax_0 = b$, the v_i corresponding to feasible components of x_0 may be fixed at zero, so that the number of infeasibilities cannot increase during subsequent iterations. In this case, if the constraints are infeasible, the optimal solution minimizes the sum of the violations of those bounds that are violated at x_0 subject to $Ax = b$. Similarly, once a component x_i becomes feasible, its corresponding violation v_i can be permanently fixed at zero. However, if the sum of the violations is to be minimized when there is no feasible point, it is necessary to allow every element of v to move.

This minimum one-norm problem is equivalent to the standard method for minimizing the sum of infeasibilities that has been used in QP and LP packages for many years. In practice, the variables u and v need not be stored explicitly, and the LP (5.1) may be solved using a variant of the simplex method in which the basis has the same dimension as that of a conventional LP with constraints $Ax = b$ and $x \geq 0$. During the solution of the LP, the search is restricted to pairs (u, v) with components satisfying $u_i \geq 0$, $v_i \geq 0$, and $u_i v_i = 0$. A feasible pair (u, v) is reconstructed from any x such that $Ax = b$. In particular, $(u_i, v_i) = (x_i, 0)$ if $x_i \geq 0$, and $(u_i, v_i) = (0, -x_i)$ if $x_i < 0$. It follows that an infeasible x_i must be kept basic because it corresponds to $(u_i, v_i) = (0, -x_i)$, with an (implicit) positive elastic variable v_i. This technique is often called *elastic programming* in the linear and nonlinear programming literature (see, e.g., Brown and Graves [8], and Gill, Murray and Saunders [38]).

The same technique can be used to find a feasible point (w, π, z) for the dual constraints $Hw - A^T\pi - z = -c$ and $z \geq 0$.

5.2 Second-Order-Consistent Basis

The nonbinding-direction methods described in Chapters 3 and 4 have the property that if the initial iterate x_0 is a subspace minimizer, then all subsequent iterates are subspace minimizers. Methods for finding an initial subspace minimizer utilize an initial estimate x_I of the QP solution, together with matrices A_B and A_N associated with an estimate of the optimal basic and nonbasic sets. These estimates are often available from the known solution of a related QP—e.g., from the solution of the previous QP subproblem in the SQP context. The initial point x_I may or may not be feasible, and the associated matrix A_B may or may not have rank m.

The definition of a second-order-consistent basis requires that the matrix A_B has rank m, so it is necessary to identify a set of linearly independent basic columns of A. One algorithm for doing this has been proposed by Gill, Murray and Saunders [38], who use a sparse LU factorization of A_B^T to identify a square nonsingular subset of the columns of A_B. If necessary, a "basis repair" scheme is used to define additional unit columns that make A_B have full rank. The nonsingular matrix B obtained as a by-product of this process may be expressed in terms of A using a column permutation P such that

$$AP = \begin{pmatrix} A_B & A_N \end{pmatrix} = \begin{pmatrix} B & S & A_N \end{pmatrix}, \tag{5.2}$$

where B is $m \times m$ and nonsingular, S is $m \times (n_B - m)$, and A_N is the $m \times n_N$ matrix consisting of the nonbasic columns of A.

The nonsingular matrix B can be used to compute a feasible point from the (possibly infeasible) initial point x_I. Given x_I, a point x_0 satisfying $Ax = b$ may be computed as

$$x_0 = x_I + P \begin{pmatrix} p_Y \\ 0 \\ 0 \end{pmatrix}, \quad \text{where} \quad Bp_Y = -(Ax_I - b).$$

The basic set $\mathcal{B}$ is second-order-consistent if the reduced KKT matrix

$$K_B = \begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix} \tag{5.3}$$

has correct inertia, i.e., n_B positive eigenvalues and m negative eigenvalues. A KKT matrix with incorrect inertia will have too many negative or zero eigenvalues. In this case, an appropriate K_B may be obtained by imposing temporary constraints that are deleted during the course of subsequent iterations. For example, if $n - m$ variables are temporarily fixed at their current values, then A_B is a square nonsingular matrix, and K_B necessarily has exactly m negative eigenvalues. The form of the temporary constraints depends on the method used to solve the reduced KKT equations (see Chapter 7).

5.2.1 Variable-reduction method

In the variable reduction method a dense Cholesky factor of the reduced Hessian $Z^T H Z$ is updated to reflect changes in the basic set (see Section 7.1). At the initial x_0 a partial Cholesky factorization with interchanges is used to find an upper-triangular matrix R that is the factor of the largest positive-definite leading submatrix of $Z^T H Z$. The use of interchanges tends to maximize the dimension of R. Let Z_R denote the columns of Z corresponding to R, and let Z be partitioned as $Z = \begin{pmatrix} Z_R & Z_A \end{pmatrix}$. A nonbasic set for which Z_R defines an appropriate null space can be obtained by fixing the variables corresponding to the columns of Z_A at their current values. As described above, minimization of $\varphi(x)$ then proceeds within the subspace defined by Z_R. If a variable is removed from the basic set, a row and column is removed from the reduced Hessian and an appropriate update is made to the Cholesky factor.

5.2.2 Schur-complement and block-LU method

If Schur-complement block-LU method is used, the procedure for finding a second-order-consistent basis is given as follows.

- Factor the reduced KKT matrix (5.3) in the form $K_B = LDL^T$, where L is a row-permuted unit lower-triangular matrix and D is block diagonal with 1×1 and 2×2 blocks (see Result 1.3.4). The inertia is determined by counting the number of positive and negative eigenvalues of D. If the inertia of K_B is correct, then we are done.

- If the inertia is incorrect, factor

$$H_A = H_B + \rho A_B^T A_B = L_A D_A L_A^T,$$

where ρ is a modest positive penalty parameter. As the inertia of K_B is not correct, D_A will have some negative eigenvalues for all positive ρ.

The factorization of H_A may be written in the form

$$H_A = L_A U \Lambda U^T L_A^T = V \Lambda V^T,$$

where $U\Lambda U^T$ is the spectral decomposition of D_A and $V = L_A U$. The block-diagonal structure of D_A implies that U is a block-diagonal orthonormal matrix.

Assume that H_A has r nonpositive eigenvalues. The inertia of Λ is the same as the inertia of H_A, and there exists a positive-semidefinite diagonal matrix E such that $\Lambda + E$ is positive definite. Since there are r nonpositive eigenvalues, E can be written in the form $E = P_r E_r P_r^T$, where E_r is an $r \times r$ diagonal matrix with positive elements and P_r is a permutation matrix such that $P_r P_r^T$ projects the diagonals of E_r into an $n_B \times n_B$ matrix. If $\bar{H}_A$ denotes the positive-definite matrix $V(\Lambda + E)V^T$, then

$$\bar{H}_A = H_A + VEV^T = H_A + VP_r E_r P_r^T V^T.$$

Define V_B as the $r \times n_B$ matrix $V_B = \frac{1}{\sqrt{\rho}} E_r^{\frac{1}{2}} P_r^T V^T$, so that

$$\bar{H}_A = H_A + \rho V_B^T V_B = H_B + \rho(A_B^T A_B + V_B^T V_B).$$

Suppose $\bar{\rho} = \gamma + \rho$ for some positive value of γ. Then, for any nonzero vector x,

$$\begin{aligned} &x^T\big(H_B + \bar{\rho}(A_B^T A_B + V_B^T V_B)\big)x \\ &\qquad = x^T\big(H_B + \rho(A_B^T A_B + V_B^T V_B)\big)x + \gamma x^T(A_B^T A_B + V_B^T V_B)x \end{aligned}$$

The first term of the above expression is positive since $\bar{H}_A$ is positive definite and the second term is nonnegative. Therefore, the matrix $H_B + \bar{\rho}(A_B^T A_B + V_B^T V_B)$ is positive definite for any $\bar{\rho} > \rho$. It follows from Debreu's Lemma 1.3.2 that the reduced Hessian $Z_B^T H Z_B$ is positive definite, where the columns of Z_B form a basis for the null space of $\begin{pmatrix} A_B \\ V_B \end{pmatrix}$. By Theorem 1.3.1, the augmented KKT matrix

$$\begin{pmatrix} H_B & A_B^T & V_B^T \\ A_B & 0 & 0 \\ V_B & 0 & 0 \end{pmatrix}$$

has "correct" inertia $(n_B, m + r, 0)$.

The minimization of $\varphi(x)$ proceeds subject to the original constraints and the (general) temporary constraints $V_B^T x_B = V_B^T (x_0)_B$, where x_0 is the initial point.

The efficiency of this scheme will depend on the number of surplus negative and zero eigenvalues in H_A. In practice, if the number of negative eigenvalues exceeds a preassigned threshold, then a temporary vertex is defined by fixing the variables associated with the columns of S in (5.2) (see Chapter 8).

5.3 Stationarity

Primal case. In the primal (standard-form) setting, a feasible x achieves stationarity if $g_B(x) = A_B^T \pi$ for some second-order-consistent basic set $\mathcal{B}$.

Suppose x_0 is feasible but not stationary, and $\mathcal{B}$ is second-order-consistent. Then x_0 can be used as the initial point for a sequence of Newton-type iterations in which $\varphi(x)$ is minimized with the nonbasic components of x fixed at their current values. Consider the equations

$$\begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix} \begin{pmatrix} p_B \\ -q_\pi \end{pmatrix} = - \begin{pmatrix} g_B - A_B^T \pi \\ 0 \end{pmatrix}.$$

These equations are the KKT equations of the equality-constrained problem

$$\underset{p \in \mathbb{R}^n}{\text{minimize}} \;\; \varphi(x_0 + p) \quad \text{subject to} \;\; Ap = 0, \quad p_N = 0. \tag{5.4}$$

Let p be the solution of (5.4). If p_B is zero (which may occur when $n_B = m$), x is a subspace stationary point (with respect to A_B) at which K_B has correct inertia. Otherwise, two situations are possible. If $x_B + p_B$ is infeasible, then feasibility is retained by determining the maximum nonnegative step $\alpha < 1$ such that $x_B + \alpha p_B$ is feasible. A variable on its bound at $x_B + \alpha p_B$ is then removed from the basic set and the iteration is repeated. The removal of a basic variable cannot increase the number of negative eigenvalues of K_B, since the removal reduces the dimension of the null space matrix Z_B by one and does not affect the positive definiteness of the reduced Hessian. Since there are a finite number of basic variables, a subspace stationary point must be determined in a finite number of steps (trivially, when enough basic variables are removed to define a vertex). If $x_B + p_B$ is feasible, then p_B is the step to the minimizer of $\varphi(x)$ with respect to the basic variables and it must hold that $g_B(x_B + p_B) = A_B^T(\pi + q_\pi)$, so that the point is a stationary point.

Dual case. Assume $\mathcal{B}$ is second-order-consistent, and that x_0 is a dual-feasible point such that $Hx_0 - A^T \pi_0 - z_0 = -c$ with $z_0 \geq 0$. Then, to reach a stationary point, a dual-feasible direction is required such that

$$x_0 + \Delta x = 0 \;\text{ and }\; A(x_0 + \Delta x) = b, \;\text{ with }\; \Delta x_N = 0. \tag{5.5}$$

Such a direction can be computed as the solution of the system:

$$\left(\begin{array}{ccc|ccc} H & 0 & 0 & Z & H & 0 \\ 0 & 0 & 0 & 0 & -A & 0 \\ 0 & 0 & 0 & 0 & -I & P_B \\ \hline Z^T & 0 & 0 & 0 & 0 & 0 \\ H & -A^T & -I & 0 & 0 & 0 \\ 0 & 0 & P_B^T & 0 & 0 & 0 \end{array}\right)\left(\begin{array}{c} \Delta w \\ \Delta\pi \\ \Delta z \\ \hline -\Delta q \\ -\Delta x \\ -\Delta y_B \end{array}\right) = -\left(\begin{array}{c} 0 \\ Ax_0 - b \\ P_N(x_0)_N \\ \hline 0 \\ 0 \\ 0 \end{array}\right).$$

This direction satisfies $Z^T\Delta w = 0$ and $H\Delta w - A^T\Delta\pi - \Delta z = 0$, so that the direction remains feasible with respect to the equality constraints of the dual problem (4.3). In addition, the third block of the system implies $\Delta x = P_B\Delta y_B - P_N(x_0)_N$, so that $\Delta x_N = -(x_0)_N$. It follows that $(x_0)_N + \Delta x_N = 0$. The second block implies $A(x_0 + \Delta x) = b$. This means the direction satisfies the conditions (5.5) required for a direction to a dual stationary point.

The defined direction can be computed from the smaller system:

$$\begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix}\begin{pmatrix} \Delta x_B \\ -\Delta\pi \end{pmatrix} = -\begin{pmatrix} -H_D(x_0)_N \\ A_B(x_0)_B - b \end{pmatrix},$$

with $\Delta x_N = -(x_0)_N$, $\Delta z_B = 0$ and $\Delta z_N = (H\Delta x - A^T\Delta\pi)_N$.

If $z + \Delta z$ is feasible, then a stationary point has been reached. If $z + \Delta z$ is not feasible, then a maximum feasible step α_F is computed, and the blocking constraint at $z + \alpha_F\Delta z \geq 0$ is removed from $\mathcal{B}$. Again, the removal of a basic variable does not affect the second-order-consistency of $\mathcal{B}$, and a stationary point will be determined in a finite number of steps.

6 Single-Phase Methods

In this chapter, the focus turns to *single-phase methods*, methods that combine the feasibility and optimality stages of the active-set method for standard-form problems

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; \varphi(x) = c^T x + \tfrac{1}{2} x^T H x \quad \text{subject to} \;\; Ax = b, \quad x \geq 0. \tag{6.1}$$

Generally, single-phase methods solve the original QP by solving a sequence of subproblems whose solutions converge to the solution of the original problem. These methods have an inner/outer iteration structure, with the outer iterations handling the updates to parameters necessary for the formulation of the subproblem, and the inner iterations being those of the method used to solve the subproblem.

Section 6.1 begins with an overview of the *penalty-function method*, leading to the derivation of two *augmented Lagrangian methods*. In Section 6.2, a more generalized approach to the augmented Lagrangian method is given from a regularization standpoint. Sections 6.3.1 and 6.3.2 consider the application of the nonbinding-direction method to the subproblems of the inner iterations, while the outer iterations are discussed in Section 6.4.

6.1 Penalty-Function Methods

Penalty-function methods are a class of methods for solving constrained problems that are not necessarily quadratic. Many choices exist for the penalty function. However, since we are interested in continuously differentiable quadratic problems, we consider the *quadratic penalty function* defined as

$$\mathcal{P}(x; \mu) = \varphi(x) + \frac{1}{2\mu} \|Ax - b\|_2^2,$$

where μ is the positive *penalty parameter*. In the "classical" penalty-function method, the smooth function $\mathcal{P}(x; \mu)$ is minimized subject to $x \geq 0$ for a sequence of decreasing values of μ. Under certain assumptions (see [28]), it can be shown that for a given sequence $\{\mu_k\}$,

$$\lim_{k \to \infty} x(\mu_k) = x^*,$$

where $x(\mu)$ is the minimizer of $\mathcal{P}(x; \mu)$ subject to $x \geq 0$, and x^* is the optimal solution of (6.1). In practice, a finite sequence of the bound-constrained subproblems is solved, with the approximate minimizer of $\mathcal{P}(x;\mu_k)$ being used as the initial estimate of the minimizer of $\mathcal{P}(x;\mu_{k+1})$.

Unfortunately, it is necessary for $\mu \to 0$ to achieve a good approximation of the QP solution. As μ decreases, the Hessian of the penalty function $\nabla^2\mathcal{P} = H + \frac{1}{\mu}A^TA$ becomes increasingly ill-conditioned, so that the subproblems become increasingly difficult to solve. To circumvent this difficulty, the equality constraints of the problem are shifted to produce a new problem that can exploit the smoothness of the quadratic penalty function and avoid the need for μ to go to zero. The shifted problem is

$$\underset{x}{\text{minimize}} \;\; \varphi(x) \quad \text{subject to} \quad Ax - s = b, \quad x \geq 0,$$

where the constant vector s defines the shifts for the equality constraints. The shifted problem is then solved by applying the penalty-function method, which leads to

$$\begin{aligned}\mathcal{P}(x;s,\mu) &= \varphi(x) + \frac{1}{2\mu}\|Ax - s - b\|_2^2 \\ &= \varphi(x) - \frac{1}{\mu}s^T(Ax-b) + \frac{1}{2\mu}\|Ax-b\|_2^2 + \frac{1}{2\mu}\|s\|_2^2.\end{aligned}$$

As s and μ are fixed parameters, the last term is irrelevant to the minimization and can be dropped. The penalty subproblem is therefore

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; \mathcal{P}(x;s,\mu) = \varphi(x) - \frac{1}{\mu}s^T(Ax-b) + \frac{1}{2\mu}\|Ax-b\|_2^2 \quad \text{subject to} \;\; x \geq 0,$$

with the gradient and Hessian of $\mathcal{P}$ given by

$$\nabla\mathcal{P}(x;s,\mu) = g(x) + \frac{1}{\mu}A^T(Ax - b - s) \quad \text{and} \quad \nabla^2\mathcal{P} = H + \frac{1}{\mu}A^TA.$$

The best choice for the shift s should make the solution of the penalty subproblem a solution of the original standard-form problem *for the current value of* μ. If $x(\mu)$ is equal to x^*, then it is necessary that $Ax(\mu) - b = 0$, and that $g(x^*) - A^T\pi^* = \nabla\mathcal{P}(x;s,\mu)$. Combined with the above expression for the gradient, these equations imply that

$$\pi^* = -\frac{1}{\mu}(Ax(\mu) - b - s) = \frac{1}{\mu}s.$$

Thus, the optimal shift is $s = \mu\pi^*$. Obviously, because the optimal multipliers are unknown, the optimal shift cannot be used to define the penalty subproblem. Therefore, s is defined as $s = \mu\pi_e$, where π_e is a vector that estimates the multipliers of $Ax = b$. With this definition, the penalty function becomes the *augmented Lagrangian function*

$$\mathcal{M}_1(x;\pi_e,\mu) = \varphi(x) - \pi_e^T(Ax-b) + \frac{1}{2\mu}\|Ax-b\|_2^2. \tag{6.2}$$

The subproblem is then

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; \mathcal{M}_1(x;\pi_e,\mu) = \varphi(x) - \pi_e^T(Ax-b) + \frac{1}{2\mu}\|Ax-b\|_2^2 \quad \text{subject to} \;\; x \geq 0, \tag{6.3}$$

which is the subproblem for the conventional *augmented Lagrangian method.*

The same approach can be applied to the bound constraints rather than the equality constraints of (6.1). As in Section 5.1, $x \geq 0$ can be rewritten as $x = u - v$, $u \geq 0$, and $v = 0$. Instead of shifting the equality constraints $Ax = b$, the constraints $v = 0$ are shifted such that $v - \mu z_e = 0$, where z_e is an estimate of the optimal multipliers for $v = 0$. This leads to the subproblem

$$\begin{array}{ll} \underset{x,u,v\in\mathbb{R}^n}{\text{minimize}} & \mathcal{M}_2(x,v;z_e,\mu) = \varphi(x) - z_e^T v + \dfrac{1}{2\mu}\|v\|_2^2 \\ \text{subject to} & Ax = b, \quad x - u + v = 0, \quad u \geq 0. \end{array} \tag{6.4}$$

Since the objective of (6.4) is a variant of the augmented Lagrangian function derived by shifting the variables v, we refer to $\mathcal{M}_2(x,v;z_e,\mu)$ as the *variable-shifted augmented Lagrangian*. For consistency, $\mathcal{M}_1(x;\pi_e,\mu)$ is the *constraint-shifted augmented Lagrangian*. The methods for solving the corresponding subproblems related to these functions are named accordingly. Also, when the values of π_e and μ are obvious from the context, they are not included as explicit arguments of the augmented Lagrangian functions, e.g., $\mathcal{M}_1(x) = \mathcal{M}_1(x;\pi_e,\mu)$.

6.2 QP Regularization

Thus far, the QP methods described have relied on the assumption that each basis matrix A_B has rank m. In an active-set method this condition is guaranteed (at least in exact arithmetic) by the active-set strategy if the *initial* basis has rank m. For methods that solve the KKT system by factoring a subset of m columns of A_B (see Section 7.1), special techniques can be used to select a linearly independent set of m columns from A. These procedures depend on the method used to factor the basis—for example, the SQP code SNOPT employs a combination of LU factorization and basis repair to determine a full-rank basis. If a factorization reveals that the square submatrix is rank deficient, suspected dependent columns are discarded and replaced by the columns associated with slack variables. However, for methods that solve the KKT system by direct factorization, such as the Schur complement method of Section 7.2, basis repair is not an option because the factor routine may be a "black-box" that does not incorporate rank-detection. Unfortunately, over the course of many hundreds of iterations, performed with KKT matrices of varying degrees of conditioning, an SQP method can place even the most robust symmetric indefinite solver under considerable stress. (Even a relatively small collection of difficult problems can test the reliability of a solver. Gould, Scott, and Hu [52] report that none of the 9 symmetric indefinite solvers tested was able to solve all of the 61 systems in their test collection.) In this situation it is necessary to use a *regularized* method, where equations are guaranteed to be solvable without the luxury of basis repair.

To illustrate how a problem may be regularized, we start by considering a QP with *equality constraints*, i.e.,

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; c^T x + \tfrac{1}{2}x^T H x \quad \text{subject to} \quad Ax = b. \tag{6.5}$$

Assume for the moment that this subproblem has a feasible primal-dual solution (x^*, π^*). Given an estimate π_e of the QP multipliers π^*, a positive μ and arbitrary ν, consider the *generalized augmented Lagrangian*

$$\mathcal{M}_3(x, \pi; \pi_e, \mu, \nu) = \varphi(x) - \pi_e^T(Ax - b) + \frac{1}{2\mu}\|Ax - b\|_2^2 + \frac{\nu}{2\mu}\|Ax - b - \mu(\pi_e - \pi)\|_2^2 \tag{6.6}$$

(see Gill and Robinson [43] for methods involving this function). The function $\mathcal{M}_3$ involves $n+m$ variables and has gradient vector

$$\nabla\mathcal{M}_3(x, \pi; \pi_e, \mu, \nu) = \begin{pmatrix} g(x) - A^T\pi + (1+\nu)A^T\big(\pi - \pi(x)\big) \\ \nu\mu\big(\pi - \pi(x)\big) \end{pmatrix}, \tag{6.7}$$

where $\pi(x) = \pi_e - (Ax - b)/\mu$. If π^* is known and π_e is defined as $\pi_e = \pi^*$, then simple substitution in (6.7) shows that (x^*, π^*) is a stationary point of $\mathcal{M}_3$ for all ν and all positive μ. The Hessian of $\mathcal{M}_3$ is given by

$$\nabla^2\mathcal{M}_3(x, \pi; \pi_e, \mu, \nu) = \begin{pmatrix} H + \big(\frac{1+\nu}{\mu}\big)A^TA & \nu A^T \\ \nu A & \nu\mu I \end{pmatrix}, \tag{6.8}$$

which is independent of π_e. If we make the additional assumptions that ν is *nonnegative* and the reduced Hessian of the QP subproblem is positive definite, then $\nabla^2\mathcal{M}_3$ is positive semidefinite for all μ sufficiently small. Under these assumptions, if $\pi_e = \pi^*$ it follows that (x^*, π^*) is the unique minimizer of the unconstrained problem

$$\underset{x\in\mathbb{R}^n, \pi\in\mathbb{R}^m}{\text{minimize}} \;\; \mathcal{M}_3(x, \pi; \pi_e, \mu, \nu). \tag{6.9}$$

This result implies that if π_e is an approximate multiplier vector (e.g., from the previous QP subproblem in the SQP context), then the minimizer of $\mathcal{M}_3(x, \pi; \pi_e, \mu, \nu)$ will approximate the minimizer of (6.5). In order to distinguish between a solution of (6.5) and a minimizer of (6.9) for an arbitrary π_e, we use (x_*, π_*) to denote a minimizer of $\mathcal{M}_3(x, \pi; \pi_e, \mu, \nu)$. Observe that stationarity of $\nabla\mathcal{M}_3$ at (x_*, π_*) implies that $\pi_* = \pi(x_*) = \pi_e - (Ax_* - b)/\mu$. The components of $\pi(x_*)$ are the so-called *first-order multipliers* associated with a minimizer of (6.9).

Particular values of the parameter ν give some well-known functions that have appeared in literature (although, as noted above, each function defines a problem with the common solution (x_*, π_*)). If $\nu = 0$, then $\mathcal{M}_3$ is independent of π, with

$$\mathcal{M}_3(x; \pi_e, \mu, 0) = \varphi(x) - (Ax - b)^T\pi_e + \frac{1}{2\mu}\|Ax - b\|_2^2 \equiv \mathcal{M}_1(x; \pi_e, \mu). \tag{6.10}$$

This is the conventional Hestenes-Powell augmented Lagrangian (6.2) introduced in Section 6.1 applied to (6.5). If $\nu = 1$ in (6.6), $\mathcal{M}_3$ is the primal-dual augmented Lagrangian

$$\varphi(x) - (Ax - b)^T\pi_e + \frac{1}{2\mu}\|Ax - b\|_2^2 + \frac{1}{2\mu}\|Ax - b - \mu(\pi_e - \pi)\|_2^2. \tag{6.11}$$

Methods for the primal-dual Lagrangian are considered in [61, 43]. If $\nu = -1$, then $\mathcal{M}_3$ is the proximal-point Lagrangian

$$\varphi(x) - (Ax - b)^T \pi - \frac{\mu}{2}\|\pi - \pi_e\|_2^2.$$

As ν is negative in this case, $\nabla^2 \mathcal{M}_3$ is indefinite and $\mathcal{M}_3$ has an unbounded minimizer. Nevertheless, a unique minimizer of $\mathcal{M}_3$ for $\nu > 0$ is a saddle-point for an $\mathcal{M}_3$ defined with a negative ν. Moreover, for $\nu = -1$, (x^*, π^*) solves the min-max problem

$$\min_x \max_\pi \;\; \varphi(x) - (Ax - b)^T \pi - \frac{\mu}{2}\|\pi - \pi_e\|_2^2.$$

In what follows, we use $\mathcal{M}_3(v)$ to denote $\mathcal{M}_3$ as a function of the primal-dual variables $v = (x, \pi)$ for given values of π_e, μ and ν. Given the initial point $v_0 = (x_0, \pi_0)$, the stationary point of $\mathcal{M}_3(v)$ is $v_* = v_0 + \Delta v$, where $\Delta v = (p, q)$ with $\nabla^2 \mathcal{M}_3(v_0)\Delta v = -\nabla \mathcal{M}_3(v_0)$. It can be shown that Δv satisfies the equivalent system

$$\begin{pmatrix} H & A^T \\ A & -\mu I \end{pmatrix} \begin{pmatrix} p \\ -q \end{pmatrix} = -\begin{pmatrix} g(x_0) - A^T \pi_0 \\ Ax_0 - b - \mu(\pi_e - \pi_0) \end{pmatrix}, \tag{6.12}$$

which is independent of the value of ν [43]. If $\nu \neq 0$, the primal-dual direction is unique. If $\nu = 0$ (i.e., $\mathcal{M}_3$ is the conventional augmented Lagrangian (6.2)), Δv satisfies the equations

$$\begin{pmatrix} H & A^T \\ A & -\mu I \end{pmatrix} \begin{pmatrix} p \\ -q \end{pmatrix} = -\begin{pmatrix} g(x_0) - A^T \pi \\ Ax_0 - b - \mu(\pi_e - \pi) \end{pmatrix}, \tag{6.13}$$

for an *arbitrary* vector π. In this case, p is unique but q depends on the choice of π. In particular, if we define the equations (6.13) with $\pi = \pi_0$, then we obtain directions identical to those of (6.12). Clearly, it must hold that p *is independent of the choice of* ν in (6.6).

The point $(x_*, \pi_*) = (x_0 + p, \pi_0 + q)$ is the primal-dual solution of the *perturbed QP*

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; c^T x + \tfrac{1}{2} x^T H x \quad \text{subject to} \;\; Ax - \mu(\pi_e - \pi_*) = b,$$

where the perturbation *shifts* each constraint of (6.5) by an amount that depends on the corresponding component of $\pi_* - \pi_e$. Observe that the constraint shift depends on the solution, so it cannot be defined *a priori*. The effect of the shift is to *regularize* the KKT equations by introducing the *nonzero* $(2, 2)$ block $-\mu I$. In the regularized case *it is not necessary for* A *to have full row rank for the KKT equations to be nonsingular*. A full-rank assumption is required if the $(2, 2)$ block is zero. In particular, if we choose $\pi_e = \pi_0$, the system (6.12) is:

$$\begin{pmatrix} H & A^T \\ A & -\mu I \end{pmatrix} \begin{pmatrix} p \\ -q \end{pmatrix} = -\begin{pmatrix} g(x_0) - A^T \pi_0 \\ Ax_0 - b \end{pmatrix}. \tag{6.14}$$

These equations define a regularized version of the Newton equations and also form the basis for the primal-dual formulations of the quadratic penalty method considered in [48] (for related methods, see Murray [57],Biggs [3] and Tapia [66]).

The price paid for the regularized equations is an *approximate* solution of the original problem. However, once (x_*, π_*) has been found, π_e can be redefined as π_* and the process repeated—with a smaller value of μ if necessary. There is more discussion of the choice of π_e below. However, before turning to the inequality constraint case, we summarize the regularization for equality constraints.

- The primal-dual solution (x^*, π^*) of the equality constraint problem (6.5) is approximated by the solution of the perturbed KKT system (6.12).

- The resulting approximation $(x_*, \pi_*) = (x_0 + p, \pi_0 + q)$ is a stationary point of the function $\mathcal{M}_3$ (6.6) regardless of the choice of ν. If $\mu > 0$ and $\nu \geq 0$ then (x_*, π_*) is a minimizer of $\mathcal{M}_3$ for all μ sufficiently small.

As the solution of the regularized problem is independent of ν, there is little reason to use nonzero values of ν in the equality-constraint case. However, the picture changes when there are *inequality constraints* and an *approximate* solution of the QP problem is required, as is often the case in the SQP context.

The method defined above can be extended to the inequality constraint problem (6.1) by solving the bound-constrained subproblem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; \mathcal{M}_1(x; \pi_e, \mu) \quad \text{subject to} \quad x \geq 0, \tag{6.15}$$

which is identical to (6.3) derived via the shifted penalty-function method. This technique has been proposed for general nonlinear programming (see, e.g., Conn, Gould and Toint [9, 10, 11], Friedlander [33], and Friedlander and Saunders [35]), and to quadratic programming (see, e.g., Dostál, Friedlander and Santos [22, 23, 24], Delbos and Gilbert [19], Friedlander and Leyffer [34]), and Maes [54]).

As in the equality-constraint case, the dual variables may be updated as $\pi_{j+1} = \pi_j + \alpha_j q_j$. The dual iterates π_j will converge to the multipliers π_* of the perturbed QP:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; c^T x + \tfrac{1}{2} x^T H x \quad \text{subject to} \quad Ax - \mu(\pi_e - \pi_*) = b, \quad x \geq 0.$$

At an optimal solution (x_*, π_*) of (6.15) the vector $z_* = g(x_*) - A^T \pi_*$ provides an estimate of the optimal reduced costs z^*. As in the equality-constraint case, the vector of first-order multipliers $\pi(x_*) = \pi_e - (Ax_* - b)/\mu$ is identical to π_*.

The algorithms defined above are *dual* regularization methods in the sense that the regularization has the effect of bounding the Lagrange multipliers. For convex QP certain *primal* regularization schemes may be used to bound the primal variables (see, e.g., Gill et al. [36], Saunders [63], Saunders and Tomlin [65, 64], Altman and Gondzio [1], and Maes [54]). The variable-shifted problem (6.4) is an example of primal regularization.

6.3 Inner Iterations

All of the methods considered above have an inner/outer iteration structure, with the outer iterations handling the updates to parameters necessary for the formulation of the subproblem, and the inner iterations being those of the method used to solve the subproblem. Next we focus on methods for solving each of the subproblems.

6.3.1 Constraint-shifted approach

If x^* and π^* satisfy the second-order sufficient conditions for the standard-form QP (6.1), then there exists a $\bar{\mu}$ such that for all $\mu < \bar{\mu}$, x^* is a solution to the constraint-shifted quadratic program (6.16) with $\pi_e = \pi^*$. This result suggests that a solution for (6.1) may be found by solving a finite sequence of problems of the form (6.3), restated here

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\;\; \mathcal{M}_1(x;\pi_e,\mu) = \varphi(x) - \pi_e^T(Ax-b) + \frac{1}{2\mu}\|Ax-b\|_2^2 \quad \text{subject to} \quad x \ge 0, \tag{6.16}$$

where π_e is an estimate of the optimal multipliers for the equality constraints $Ax = b$. The optimality conditions are given in terms of the gradient of $\mathcal{M}_1$,

$$\nabla\mathcal{M}_1(x;\pi_e,\mu) = g(x) - A^T\Big(\pi_e - \frac{1}{\mu}(Ax-b)\Big) = g(x) - A^T\pi(x),$$

with $\pi(x)$ defined as the vector $\pi(x) = \pi_e - \frac{1}{\mu}(Ax-b)$, and the Hessian of $\mathcal{M}_1$, $\nabla^2\mathcal{M}_1 = H + \frac{1}{\mu}A^TA$.

Result 6.3.1 (Optimality conditions). *If x^* is a local minimizer of the QP* (6.16)*, then*

(a) $x^* \ge 0$*;*

(b) *there exists a vector z^* such that $\nabla\mathcal{M}_1(x^*;\pi_e,\mu) = z^*$ with $z^* \ge 0$;*

(c) $x^* \cdot z^* = 0$*; and*

(d) $p^T\nabla^2\mathcal{M}_1 p \ge 0$ *for all p such that $p_i \ge 0$ if $x_i^* = 0$.*

The first-order optimality conditions for (6.16) may be written equivalently in active-set form. Let $P_{\mathfrak{a}}^T$ denote the active-set matrix at x^*. Conditions (b) and (c) of Result 6.3.1 are equivalent to

$$\nabla\mathcal{M}_1(x^*) = P_{\mathfrak{a}}\, z_{\mathfrak{a}}\,, \;\; \text{where} \;\; z^* = P_{\mathfrak{a}}\, z_{\mathfrak{a}} \;\; \text{with} \;\; z_{\mathfrak{a}} \ge 0.$$

Result 6.3.2. *If x^* satisfies the second-order sufficient conditions of* (6.1) *with strict complementarity, then there exists a $\bar{\mu}$ such that x^* is a solution to* (6.16) *for all μ such that $0 < \mu \le \bar{\mu}$.*

Proof. Let $P_{\mathfrak{a}}^T$ denote the active-set matrix at x^*. The first-order sufficient conditions for (6.1) imply $Ax^* = b$, $x^* \ge 0$, and $g(x^*) = A^T\pi^* + P_{\mathfrak{a}}\, z_{\mathfrak{a}}$ with $z_{\mathfrak{a}} > 0$. Thus, the feasibility condition for (6.16) is satisfied because $x \ge 0$.

Let $\pi_e = \pi^*$. Then

$$\begin{aligned}\nabla\mathcal{M}_1(x^*) &= g(x^*) - A^T\big(\pi_e - \frac{1}{\mu}(Ax^* - b)\big)\\ &= g(x^*) - A^T\pi_e \qquad (\text{since } Ax^* = b)\\ &= P_{\mathfrak{a}}\, z_{\mathfrak{a}}\,,\end{aligned}$$

so that the stationarity condition for (6.16) is satisfied.

The second-order sufficient conditions for (6.1) imply that there exists an $\omega > 0$ such that

$$p^T H p \geq \omega\|p\|_2^2 \quad \text{for all } \ p \in \text{null}\begin{pmatrix} A \\ P_{\mathfrak{a}}^T \end{pmatrix}.$$

By Debreu's Lemma 1.3.2, this condition holds if and only if there exists a $\bar{\mu} > 0$ such that $H + \frac{1}{\mu}A^TA$ is positive definite for all $0 < \mu \leq \bar{\mu}$. Thus, the sufficient conditions for (6.16) are satisfied and x^* is a solution of (6.16) with $\pi_e = \pi^*$. □

Application of the nonbinding-direction method to the constraint-shifted approach resembles the standard-form version of the method. The working set is the nonbasic set $\mathcal{N}$, with corresponding working-set matrix P_N^T composed of unit columns $\{e_i\}$ with $i \in \mathcal{N}$. The complementary basic set $\mathcal{B}$ defines the matrix P_B^T. Unlike the other algorithms described, no assumption on the rank of A_B is required.

Result 6.3.3 (Subspace minimizer)**.** *Let $\mathcal{B}$ be the basic set for a point $x \geq 0$. Then*

(a) *If x is a* subspace stationary point with respect to $\mathcal{B}$*, then $g_B(x) = A_B^T\pi(x)$.*

(b) *If $\mathcal{B}$ is a* second-order-consistent basis *for the problem* (6.16)*, then the KKT matrix*

$$\begin{pmatrix} H_B & A_B^T \\ A_B & -\mu I \end{pmatrix} \tag{6.17}$$

has inertia $(n_B, m, 0)$.

Proof. By definition, a stationary point is a point where the gradient of the objective lies in the range space of the transpose of the working-set matrix. Thus, there exists a vector z_N such that $\nabla\mathcal{M}_1(x;\pi_e,\mu) = P_N z_N$. This implies that $z_N = g_N(x) - A_N^T\pi(x)$ and $0 = g_B(x) - A_B^T\pi(x)$.

A second-order-consistent basis for subproblem (6.16) implies that the KKT matrix

$$\begin{pmatrix} \nabla^2\mathcal{M}_1 & P_N \\ P_N^T & 0 \end{pmatrix}$$

has inertia $(n, n_N, 0)$, or equivalently, that $Z^T\nabla^2\mathcal{M}_1 Z$ is positive definite, where the columns of Z form a basis for the null space of P_N^T. However, since P_N^T is a permutation matrix, Z can be defined as $Z = P_B$. Thus

$$Z^T\nabla^2\mathcal{M}_1 Z = P_B^T(H + \frac{1}{\mu}A^TA)P_B = H_B + \frac{1}{\mu}A_B^TA_B.$$

Then by Theorem 1.3.2, the shifted KKT-matrix (6.17) has inertia $(n_B, m, 0)$. □

Once a negative multiplier $z_{\nu_s} = [z_N]_s$ is identified, the search direction is defined as the solution of

$$\begin{pmatrix} \nabla^2\mathcal{M}_1 & P_N \\ P_N^T & 0 \end{pmatrix} \begin{pmatrix} p \\ -q_N \end{pmatrix} = \begin{pmatrix} 0 \\ e_s \end{pmatrix}. \tag{6.18}$$

If we define the auxiliary vector $q_\pi = -\frac{1}{\mu}Ap$, the first block of this system becomes

$$0 = \Big(H + \frac{1}{\mu}A^TA\Big)p - P_N q_N = Hp - A^T q_\pi - P_N q_N.$$

The KKT system can be rewritten to include the components of q_π as unknowns, giving the equivalent system

$$\begin{pmatrix} H & A^T & P_N \\ A & -\mu I & \\ P_N^T & & \end{pmatrix} \begin{pmatrix} p \\ -q_\pi \\ -q_N \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ e_s \end{pmatrix}.$$

If this system is reduced to its basic components as in Chapter 3, the solution of (6.18) can be computed from

$$\begin{pmatrix} H_B & A_B^T \\ A_B & -\mu I \end{pmatrix} \begin{pmatrix} p_B \\ -q_\pi \end{pmatrix} = -\begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} \\ a_{\nu_s} \end{pmatrix}, \tag{6.19}$$

with $p_N = e_s$ and $q_N = (Hp - A^T q_\pi)_{\mathcal{N}}$. Apart from the $-\mu I$ term in the KKT matrix, these equations are identical to System 1 of the standard-form nonbinding-direction method (3.4).

A simple calculation gives the identities

$$\nabla\mathcal{M}_1^T p = p^T P_N^T z_N = [z_N]_s \quad \text{and} \quad p^T \nabla^2\mathcal{M}_1 p = p^T P_N q_N = [q_N]_s,$$

from which the optimal step may be calculated as $\alpha_* = -[z_N]_s/[q_N]_s$. The feasible step is identical to that defined in (3.7), since the standard-form problem has the same inequality bounds $x \geq 0$.

If the optimal step is taken, then ν_s can be added to $\mathcal{B}$. Otherwise, if the problem is bounded, there must be a blocking constraint $x_{\beta_r} \geq 0$ at $x + \alpha p$. In this case, the second KKT system for the constraint-shifted problem is given by

$$\begin{pmatrix} H_B & A_B^T \\ A_B & -\mu I \end{pmatrix} \begin{pmatrix} u_B \\ -v_\pi \end{pmatrix} = \begin{pmatrix} e_r \\ 0 \end{pmatrix}, \tag{6.20}$$

which may be derived in the same way as (6.19) above. If $u_B = 0$, then the second block of equations in (6.20) implies that $v_\pi = 0$. However, this implies a contradiction because the right-hand side of the first block of equations of (6.20) is nonzero. Thus, u_B cannot be zero and a blocking constraint can be removed from $\mathcal{B}$ (and added to $\mathcal{N}$) immediately by parts (2a) and (2c) of Result 3.2.2. Since it is always permissible to add a blocking constraint, there is no need to solve (6.20). This result may also be inferred from the fact that the working-set matrix consists of rows of the identity, and any blocking constraint is linearly independent of the rows of P_N^T.

The following result summarizes the updates to the basic set. Proofs are omitted as they are almost identical to those found in Chapter 3.

Result 6.3.4 (Basic set updates)**.** *Let x be a subspace minimizer with respect to $\mathcal{B}$ and let p and q be defined by* (6.19)*. Define $\bar{x} = x + \alpha p$.*

(a) *If $\alpha = \alpha_*$, then $\bar{x}$ is a subspace minimizer with respect to $\bar{\mathcal{B}} = \mathcal{B} + \{\nu_s\}$.*

(b) *If $x_{\beta_r} \geq 0$ is a blocking constraint at $\bar{x}$, then $\bar{x}$ is a subspace minimizer with respect to $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\}$.*

There are two obvious benefits to the constraint-shifted method. First, there is no need to find an initial point such that $Ax = b$. Second, it is necessary to solve only one KKT system at each iteration, which implies that there is no advantage to updating p and q as in the conventional nonbinding direction method.

Algorithm 6.1 summarizes the method. As before, `minRatioTest` computes the maximum feasible step, and `subspaceMin` returns a subspace minimizer. In Algorithm 6.1, the multiplier z is computed explicitly at each iteration rather than being updated.

6.3.2 Variable-shifted approach

The subproblem of the variable-shifted method is

$$\begin{array}{ll} \underset{x,u,v\in\mathbb{R}^n}{\text{minimize}} & \mathcal{M}_2(x,v;z_e,\mu) = \varphi(x) - z_e^T v + \dfrac{1}{2\mu}\|v\|_2^2 \\ \text{subject to} & Ax = b, \quad x - u + v = 0, \quad u \geq 0, \end{array} \tag{6.21}$$

where μ is the positive penalty parameter and z_e is a constant estimate of the multiplier vector for the constraints $v = 0$. The gradient and Hessian of the objective function are given by

$$\nabla\mathcal{M}_2(x,v;z_e,\mu) = \begin{pmatrix} g(x) \\ 0 \\ \frac{1}{\mu}v - z_e \end{pmatrix}, \text{ and } \nabla^2\mathcal{M}_2 = \begin{pmatrix} H & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\mu}I \end{pmatrix}.$$

The first-order stationarity condition for this problem implies that there exist vectors π, z and z_u such that

$$\begin{pmatrix} g(x) \\ 0 \\ \frac{1}{\mu}v - z_e \end{pmatrix} = \begin{pmatrix} A^T & I \\ & -I \\ & I \end{pmatrix} \begin{pmatrix} \pi \\ y \end{pmatrix} + \begin{pmatrix} 0 \\ z \\ 0 \end{pmatrix},$$

with non-negativity and complementarity conditions $z \geq 0$ and $z \cdot u = 0$. Together, these conditions imply that

$$g(x) = A^T\pi + z, \quad z \geq 0, \quad z \cdot u = 0, \quad \text{and} \quad z = \frac{1}{\mu}v - z_e.$$

Algorithm 6.1: Constraint-shifted algorithm

Find x_0 such that $x_0 \geq 0$;

$[x, \mathcal{B}] = \texttt{subspaceMin}(x_0)$;

$\pi \leftarrow \pi_e - \frac{1}{\mu}(Ax - b)$; $\quad z \leftarrow c + Hx - A^T\pi$;

$\nu_s = \text{argmin}_i\{z_i\}$;

while $z_{\nu_s} < 0$ **do**

  Solve $\begin{pmatrix} H_B & A_B^T \\ A_B & -\mu I \end{pmatrix} \begin{pmatrix} p_B \\ -q_\pi \end{pmatrix} = -\begin{pmatrix} (h_{\nu_s})_{\mathcal{B}} \\ a_{\nu_s} \end{pmatrix}$; $\quad p = P\begin{pmatrix} p_B \\ e_s \end{pmatrix}$; $\quad q_N = (Hp - A^T q_\pi)_{\mathcal{N}}$;

  $\alpha_F = \texttt{minRatioTest}(x_B, p_B)$;

  if $[q_N]_s > 0$ **then** $\alpha_* = -z_{\nu_s}/[q_N]_s$ **else** $\alpha_* = +\infty$;

  $\alpha = \min\{\alpha_*, \alpha_F\}$;

  if $\alpha = +\infty$ **then stop**; [the solution is unbounded]

  $x \leftarrow x + \alpha p$; $\quad \pi \leftarrow \pi + \alpha q_\pi$; $\quad z \leftarrow c + Hx - A^T\pi$;

  if $\alpha_F < \alpha_*$ **then** [remove r-th basic variable]

    Find the blocking constraint index r;

    $\mathcal{B} \leftarrow \mathcal{B} - \{\beta_r\}$;

  else [add s-th nonbasic variable]

    $\mathcal{B} \leftarrow \mathcal{B} + \{\nu_s\}$;

    $\nu_s = \text{argmin}_i\{z_i\}$;

  end;

  $k \leftarrow k + 1$;

end while

The working-set indices of u are denoted by $\mathcal{N}$. The working-set matrix A_w is defined as

$$A_w = \begin{pmatrix} A & 0 & 0 \\ I & -I & I \\ 0 & P_N^T & 0 \end{pmatrix}, \tag{6.22}$$

where P_N^T is defined by $\mathcal{N}$. As usual, the basic set $\mathcal{B}$ is the complementary set of indices such that $\mathcal{B} \cup \mathcal{N} = \{1, \ldots, n\}$.

In the next result, we derive the properties of a subspace stationary point and a second-order-consistent basis in terms relevant to the variable-shifted algorithm. In particular, the matrix

$$K = \left(\begin{array}{cc|c} H & A^T & P_N \\ A & 0 & 0 \\ \hline P_N^T & 0 & -\mu I \end{array}\right),$$

which appears in the equations solved in the algorithm, is shown to have a specific inertia.

Result 6.3.5 (Subspace minimizer)**.** *Let (x, u, v) be a feasible point with basic and nonbasic sets $\mathcal{B}$ and $\mathcal{N}$.*

(a) *If (x, u, v) is a subspace stationary point with respect to $\mathcal{B}$, then $g_{\scriptscriptstyle B}(x) = A_{\scriptscriptstyle B}^T\pi$, and $z_{\scriptscriptstyle N} = (\frac{1}{\mu}v - z_e)_{\mathcal{N}}$.*

(b) *If $\mathcal{B}$ is a second-order-consistent basis for* (6.21)*, then the matrix*

$$K = \left(\begin{array}{cc|c} H & A^T & P_{\scriptscriptstyle N} \\ A & 0 & 0 \\ \hline P_{\scriptscriptstyle N}^T & 0 & -\mu I \end{array}\right) \tag{6.23}$$

has inertia $(n, m + n_{\scriptscriptstyle N}, 0)$.

Proof. By definition, $\nabla\mathcal{M}_2(x, v; z_e, \mu)$ lies in the range-space of the transpose of the working-set matrix (6.22). Thus, there exist vectors π, y and $z_{\scriptscriptstyle N}$ such that

$$\begin{pmatrix} g(x) \\ 0 \\ \frac{1}{\mu}v - z_e \end{pmatrix} = \begin{pmatrix} A^T & I & 0 \\ 0 & -I & P_{\scriptscriptstyle N} \\ 0 & I & 0 \end{pmatrix} \begin{pmatrix} \pi \\ y \\ z_{\scriptscriptstyle N} \end{pmatrix}.$$

This implies that $g(x) = A^T\pi + P_{\scriptscriptstyle N} z_{\scriptscriptstyle N}$ and $P_{\scriptscriptstyle N} z_{\scriptscriptstyle N} = \frac{1}{\mu}v - z_e$. Therefore, at a subspace stationary point, it holds that

$$g_{\scriptscriptstyle B}(x) = A_{\scriptscriptstyle B}^T\pi, \;\text{ with }\; z_{\scriptscriptstyle B} = 0 \quad\text{and}\quad z_{\scriptscriptstyle N} = (\frac{1}{\mu}v - z_e)_{\mathcal{N}}.$$

For part (b), we use Theorem 1.3.2 to relate the inertia of K to the inertia of the reduced Hessian matrix $Z^T(H + \frac{1}{\mu}P_{\scriptscriptstyle N}P_{\scriptscriptstyle N}^T)Z$, where the columns of Z form a basis for the null space of A. If "H" is the matrix $\begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix}$ and "A" is $\begin{pmatrix} P_{\scriptscriptstyle N}^T & 0 \end{pmatrix}$ in Theorem 1.3.2, then the theorem states that

$$\begin{aligned} \operatorname{In}(K) &= \operatorname{In}\Big(\begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix} + \frac{1}{\mu}\begin{pmatrix} P_{\scriptscriptstyle N} \\ 0 \end{pmatrix}\begin{pmatrix} P_{\scriptscriptstyle N}^T & 0 \end{pmatrix} \Big) + (0, n_{\scriptscriptstyle N}, 0) \\ &= \operatorname{In}\begin{pmatrix} H + \frac{1}{\mu}P_{\scriptscriptstyle N}P_{\scriptscriptstyle N}^T & A^T \\ A & 0 \end{pmatrix} + (0, n_{\scriptscriptstyle N}, 0) \\ &= \operatorname{In}(Z^T(H + \frac{1}{\mu}P_{\scriptscriptstyle N}P_{\scriptscriptstyle N}^T)Z) + (m, m, 0) + (0, n_{\scriptscriptstyle N}, 0), \end{aligned}$$

where the last equality holds from Corollary 1.3.1. Therefore, it is sufficient to show that the $(n - m) \times (n - m)$ reduced Hessian $Z^T(H + \frac{1}{\mu}P_{\scriptscriptstyle N}P_{\scriptscriptstyle N}^T)Z$ is positive definite, and hence has inertia $(n - m, 0, 0)$.

Define Q such that

$$Q = \begin{pmatrix} Z & 0 \\ P_{\scriptscriptstyle B}P_{\scriptscriptstyle B}^T Z & P_{\scriptscriptstyle B} \\ -P_{\scriptscriptstyle N}P_{\scriptscriptstyle N}^T Z & P_{\scriptscriptstyle B} \end{pmatrix}.$$

This matrix has linearly independent columns. For any vector u in the null space of Q, it must hold that

$$0 = Qu = \begin{pmatrix} Z & 0 \\ P_{\scriptscriptstyle B} P_{\scriptscriptstyle B}^T Z & P_{\scriptscriptstyle B} \\ -P_{\scriptscriptstyle N} P_{\scriptscriptstyle N}^T Z & P_{\scriptscriptstyle B} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix},$$

which implies that $Zu_1 = 0$ and $P_{\scriptscriptstyle B} u_2 = 0$, and hence $u = 0$. Also, since $A_w Q = 0$, the columns of Q form a basis for the null space for A_w (6.22). By definition of a second-order-consistent basis, the matrix $Q^T \nabla^2 \mathcal{M}_2 Q$ must be positive definite. If the terms of Q and the Hessian are expanded, then

$$\begin{aligned} Q^T \nabla^2 \mathcal{M}_2 Q &= \begin{pmatrix} Z^T H Z + \frac{1}{\mu} Z^T P_{\scriptscriptstyle N} P_{\scriptscriptstyle N}^T P_{\scriptscriptstyle N} P_{\scriptscriptstyle N}^T Z & -\frac{1}{\mu} Z^T P_{\scriptscriptstyle N} P_{\scriptscriptstyle N}^T P_{\scriptscriptstyle B} \\ -\frac{1}{\mu} P_{\scriptscriptstyle B}^T P_{\scriptscriptstyle N} P_{\scriptscriptstyle N}^T Z & \frac{1}{\mu} P_{\scriptscriptstyle B}^T P_{\scriptscriptstyle B} \end{pmatrix} \\ &= \begin{pmatrix} Z^T H Z + \frac{1}{\mu} Z^T P_{\scriptscriptstyle N} P_{\scriptscriptstyle N}^T Z & 0 \\ 0 & \frac{1}{\mu} I_B \end{pmatrix} \\ &= \begin{pmatrix} Z^T (H + \frac{1}{\mu} P_{\scriptscriptstyle N} P_{\scriptscriptstyle N}^T) Z & 0 \\ 0 & \frac{1}{\mu} I_B \end{pmatrix}. \end{aligned}$$

Since the leading principal submatrix of a symmetric positive-definite matrix is positive definite, $Z^T(H + \frac{1}{\mu} P_{\scriptscriptstyle N} P_{\scriptscriptstyle N}^T)Z$ is also positive definite. Therefore, K has inertia $(n, m + n_{\scriptscriptstyle N}, 0)$. $\square$

If $z_{\nu_s} < 0$, a search direction is computed by solving the system:

$$\left(\begin{array}{ccc|ccc} H & 0 & 0 & A^T & I & 0 \\ 0 & 0 & 0 & 0 & -I & P_{\scriptscriptstyle N} \\ 0 & 0 & \frac{1}{\mu} I & 0 & I & 0 \\ \hline A & 0 & 0 & 0 & 0 & 0 \\ I & -I & I & 0 & 0 & 0 \\ 0 & P_{\scriptscriptstyle N}^T & 0 & 0 & 0 & 0 \end{array}\right) \left(\begin{array}{c} p_x \\ p_u \\ p_v \\ \hline -q_\pi \\ -y \\ -q_N \end{array}\right) = \left(\begin{array}{c} 0 \\ 0 \\ 0 \\ \hline 0 \\ 0 \\ e_s \end{array}\right). \tag{6.24}$$

The second and third blocks of these equations indicate that $y = \frac{1}{\mu} p_v = P_N q_N$, so that $\frac{1}{\mu}(p_v)_N = q_N$ and $(p_v)_B = 0$. The sixth block implies $(p_u)_N = e_s$. Combined with the fifth block, $p_x - p_u + p_v = 0$, we get that

$$(p_x)_N = e_s - (p_v)_N = e_s - \mu q_N \;\text{ and }\; (p_x)_B = (p_u)_B.$$

The remaining equations imply

$$Hp_x - A^T q_\pi - P_N q_N = 0, \quad Ap_x = 0, \quad P_N^T p_x = (p_x)_N = e_s - \mu q_N,$$

which can be combined to form the variable-shifted version of System 1:

$$\begin{pmatrix} H & A^T & P_{\scriptscriptstyle N} \\ A & 0 & 0 \\ P_{\scriptscriptstyle N}^T & 0 & -\mu I \end{pmatrix} \begin{pmatrix} p_x \\ -q_\pi \\ -q_N \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ e_s \end{pmatrix}, \tag{6.25}$$

with $p_v = \mu P_N q_N$ and $p_u = p_x + p_v$.

Based on previous experience, it may seem possible to reduce (6.25) further by decomposing it into its basic and nonbasic components. However, the $-\mu I$ term in the $(3,3)$ block does not allow for this possibility and the algorithm must solve the system with a KKT matrix that includes the entire Hessian H and constraint matrix A.

If p is partitioned as (p_x, p_u, p_v), it must hold that

$$p^T \nabla^2 \mathcal{M}_2 p = p_x^T H p_x + \frac{1}{\mu} p_v^T p_v = (e_s - \mu q_N)^T q_N + \mu q_N^T q_N = [q_N]_s,$$

because of the equations in System 1 (6.25). Moreover,

$$\begin{aligned}
\nabla \mathcal{M}_2^T p &= g^T p_x + (\frac{1}{\mu} v - z_e)^T p_v \\
&= g^T p_x + z_N^T (p_v)_N \qquad \text{(by the stationarity condition and the identity } (p_v)_B = 0) \\
&= (g - A^T \pi)^T p_x + z_N^T (p_v)_N \qquad (\text{since } A p_x = 0) \\
&= z_N^T (p_x)_N + z_N^T (e_s - (p_x)_N) \qquad (\text{since } z_B = 0) \\
&= [z_N]_s.
\end{aligned}$$

Therefore, the optimal step for the variable-shifted problem is defined as $\alpha_* = -[z_N]_s/[q_N]_s$. The feasible step is computed as in (3.7), except that x is replaced by u (since the bounds of (6.21) are $u \geq 0$).

Once the step and direction are known, the updates to the variables and multipliers are

$$x + \alpha p_x, \quad u + \alpha p_u, \quad v + \alpha p_v \quad \text{and,} \quad \pi + \alpha q_\pi \; \text{ and } \; z + \alpha P_N q_N.$$

If $\alpha = \alpha_*$, then the next working set is $\mathcal{B} + \{\nu_s\}$. Otherwise, if $\alpha = \alpha_F$, a blocking constraint β_r is removed from the basic set, $\mathcal{B} - \{\beta_r\}$ and a second system

$$\left(\begin{array}{ccc|ccc}
H & 0 & 0 & A^T & I & 0 \\
0 & 0 & 0 & 0 & -I & P_N \\
0 & 0 & \frac{1}{\mu} I & 0 & I & 0 \\
\hline
A & 0 & 0 & 0 & 0 & 0 \\
I & -I & I & 0 & 0 & 0 \\
0 & P_N^T & 0 & 0 & 0 & 0
\end{array}\right)
\left(\begin{array}{c}
u_x \\ u_u \\ u_v \\ \hline -v_\pi \\ -y \\ -v_N
\end{array}\right)
=
\left(\begin{array}{c}
0 \\ e_{\beta_r} \\ 0 \\ \hline 0 \\ 0 \\ 0
\end{array}\right) \tag{6.26}$$

is solved to determine constraint dependency in the working-set matrix. By Result 2.2.6, the blocking constraint is linearly dependent on the working-set constraints if and only if $u_x = u_u = u_v = 0$. However, the second and third blocks of (6.26) imply that $u_v = \mu(e_{\beta_r} + P_N v_N)$, so that $(u_v)_{\mathcal{B}} = \mu e_r \neq 0$. Thus, a blocking constraint is always linearly independent of the working-set constraints. This can also be seen in the structure of the working-set matrix (6.22). Since A is assumed to have rank m and $\text{rank}(A_w) = \text{rank}(A) + n + n_N$. the working-set matrix trivially has full-rank. For completeness, however, we note that System 2 for the variable-shifted method is

$$\begin{pmatrix} H & A^T & P_N \\ A & 0 & 0 \\ P_N^T & 0 & -\mu I \end{pmatrix} \begin{pmatrix} u_x \\ -v_\pi \\ -v_N \end{pmatrix} \begin{pmatrix} e_{\beta_r} \\ 0 \\ 0 \end{pmatrix}, \tag{6.27}$$

with $u_v = \mu(P_N v_N + e_{\beta_r})$ and $u_u = u_x + u_v$.

It only remains to show that the updated variables and basic sets define a subspace minimizer.

Result 6.3.6. *Let (x, u, v) be a subspace minimizer with respect to $\mathcal{B}$. Assume the solution of (6.25) has been computed and let $(\bar{x}, \bar{u}, \bar{v}) = (x, u, v) + \alpha(p_x, p_u, p_v)$, $\bar{\pi} = \pi + \alpha q_\pi$, and $\bar{z} = z + \alpha P_N q_N$.*

(a) *If $\alpha = \alpha_*$, then $(\bar{x}, \bar{u}, \bar{v})$ is a subspace minimizer with respect to $\bar{\mathcal{B}} = \mathcal{B} + \{\nu_s\}$.*

(b) *If $\alpha = \alpha_F$, then $(\bar{x}, \bar{u}, \bar{v})$ is a subspace minimizer with respect to $\bar{\mathcal{B}} = \mathcal{B} - \{\beta_r\}$, where β_r is a blocking constraint at $\bar{u}$.*

Proof. We show that the parallel subspace property holds.

$$g(\bar{x}) - A^T\bar{\pi} = g(x) - A^T\pi + \alpha(Hp_x - A^T q_\pi) = z + \alpha P_N q_N = \bar{z}.$$

Thus, $\bar{z}_B = 0$. Also because $p_v = \mu P_N q_N$, it holds that $\bar{z}_N = (\frac{1}{\mu}(v + p_v) - z_e)_{\mathcal{N}}$. Since this identity applies for any scalar α, $(\bar{x}, \bar{u}, \bar{v})$ remains a subspace stationary point in both cases.

The proof for part (a) is almost identical to the analogous proof in Result 2.2.5. The only difference is the existence of $-\mu$ in the $(1, 1)$ position of $\widetilde{M}$ defined in (2.21), but this causes no difficulties and (2.22) still holds. The remainder of the proof follows "as is".

For part (b), the permuted KKT matrix for $\bar{\mathcal{B}}$ is

$$\bar{K} = \left(\begin{array}{ccc|c} H & A^T & P_N & e_{\beta_r} \\ A & 0 & 0 & 0 \\ P_N^T & 0 & -\mu I & 0 \\ \hline e_{\beta_r}^T & 0 & 0 & -\mu \end{array}\right).$$

The Schur complement matrix is given by $\bar{K}/K = -\mu - e_{\beta_r}^T K^{-1} e_{\beta_r} = -(\mu + e_{\beta_r}^T u_x)$. The following argument may be used to verify that $e_{\beta_r}^T u_x > 0$. Using System 2 (6.27), we have

$$\begin{aligned} u_x^T e_{\beta_r} = u_x^T(Hu_x - A^T v_\pi - P_N v_N) &= u_x^T H u_x - u_x^T P_N v_N \\ &= u_x^T H u_x + \frac{1}{\mu} u_x^T P_N P_N^T u_x = u_x^T(H + \frac{1}{\mu} P_N P_N^T) u_x. \end{aligned}$$

As $Au_x = 0$, and the matrix $H + \frac{1}{\mu} P_N P_N^T$ is positive definite on the null space of A by part (b) of Result 6.3.5, it follows that $e_{\beta_r}^T u_x > 0$. Thus, $\bar{K}/K < 0$ and $\mathrm{In}(\bar{K}) = \mathrm{In}(K) + (0, 1, 0) = (n, m + n_N + 1, 0)$, as required. □

Algorithm 6.2: Variable-shifted algorithm

Find x such that $Ax = b$;

Define u and v such that $u_i = \max\{x_i, 0\}$ and $v_i = \min\{v_i, 0\}$;

$[x, \pi, z, \mathcal{B}] = \texttt{subspaceMin}(x)$;

$\nu_s = \text{argmin}_i\{z_i\}$;

while $z_{\nu_s} < 0$ **do**

  Solve $\begin{pmatrix} H & A^T & P_N \\ A & 0 & 0 \\ P_N^T & 0 & -\mu I \end{pmatrix} \begin{pmatrix} p_x \\ -q_\pi \\ -q_N \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ e_s \end{pmatrix}$; $\quad p_v = \mu P_N q_N; \quad p_u = p_x + p_v$;

  $\alpha_F = \texttt{minRatioTest}(u_B, (p_u)_B)$;

  if $[q_N]_s > 0$ **then** $\alpha_* = -z_{\nu_s}/[q_N]_s$ **else** $\alpha_* = +\infty$;

  $\alpha = \min\{\alpha_*, \alpha_F\}$;

  if $\alpha = +\infty$ **then stop**; [the solution is unbounded]

  $x \leftarrow x + \alpha p; \quad \pi \leftarrow \pi + \alpha q_\pi; \quad z \leftarrow z + \alpha q_N$;

  if $\alpha_F < \alpha_*$ **then** [add r-th basic variable]

    Find the blocking constraint index r;

    $\mathcal{B} \leftarrow \mathcal{B} - \{\beta_r\}$;

  else [remove s-th nonbasic variable]

    $\mathcal{B} \leftarrow \mathcal{B} + \{\nu_s\}$;

    $\nu_s = \text{argmin}_i\{z_i\}$;

  end;

  $k \leftarrow k + 1$;

end while

It is noted that the solution of a generic KKT system of the form

$$\begin{pmatrix} H & A^T & P_N \\ A & 0 & 0 \\ P_N^T & 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix},$$

can be computed from the smaller system

$$\begin{pmatrix} H + \frac{1}{\mu} P_N P_N^T & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a + \frac{1}{\mu} P_N c \\ b \end{pmatrix},$$

with $z = \frac{1}{\mu}(P_N^T x - c)$. However, within a QP algorithm, the latter KKT matrix is difficult to maintain since the $(1,1)$ block is in terms of the matrix P_N, which changes at every iteration.

6.4 Outer Iterations

In order to get a solution for the standard-form QP (6.1) using a single-phase methods described in the previous section, a sequence of constraint-shifted (6.16) or the variable-shifted (6.21) subproblems needs to be solved with decreasing values of μ. In this section, the updates to the penalty parameter and the multiplier estimates that occur in the outer iterations are addressed. However, the discussion is limited to describing an algorithm for the single-phase method involving the constraint-shifted subproblem (6.16). The discussion can be extended to methods using the variable-shifted subproblem (6.21).

If the QP is a "one-off" problem, then established techniques associated with the bound-constrained augmented Lagrangian method can be used to update π_e and μ (see, e.g., Conn, Gould and Toint [10], Dostál, Friedlander and Santos [22, 23, 24], Delbos and Gilbert [19], Friedlander and Leyffer [34], and Maes [54]). These rules are designed to update π_e and μ without the need to find the exact solution of (6.16). In the SQP context, it may be more appropriate to find an approximate solution of (6.16) for a *fixed* value of π_e, which is then updated in the outer iteration. Moreover, as μ is being used principally for regularization, it is given a smaller value than is typical in a conventional augmented Lagrangian method.

For the constraint-shifted problem (6.16), we apply the bound-constrained Lagrangian (BCL) method considered in Friedlander [33], where global convergence results can be found. The algorithm is given in Algorithm 6.3. The multiplier estimates π_e are denoted by π_k, where k is the outer iteration count. Other quantities are also given a subscript k to denote their values at the k-th iteration. Updates to the multiplier estimates and the penalty parameter μ_k are determined by the solution of the subproblem, denoted by x_k^* with multipliers z_k^*. If $\|Ax_k^* - b\| > \max\{\eta_k, \eta_*\}$ for some convergence tolerances η_k and η_*, then the multiplier estimates are not updated, and the penalty parameter is decreased. Otherwise, the value of π_k is updated, and the penalty parameter is unchanged. Furthermore, if the subproblem solution is deemed optimal for the original standard-form QP, then the algorithm terminates.

The BCL algorithm can solve the bound-constrained subproblem of the inner iterations inexactly, without impeding convergence. The first-order optimality conditions in Result 6.3.1 of (6.16) are relaxed to give approximate conditions

$$x \geq 0, \tag{6.28a}$$

$$z \geq -\omega_k e, \tag{6.28b}$$

$$z = \nabla \mathcal{M}_1(x; \pi_k, \rho), \tag{6.28c}$$

$$x \cdot z \leq \omega_k e, \tag{6.28d}$$

where $\omega_k \geq 0$ is the k-th optimality tolerance. These conditions are used as a stopping criteria for solving the subproblems. Similar stopping criteria are given for the BCL algorithm. A point (x, π, z) is deemed optimal for (6.1), if it satisfies the relaxed first-order optimality conditions of

(6.1), given as

$$x \geq 0, \tag{6.29a}$$

$$z \geq -\omega_* e, \tag{6.29b}$$

$$z = \nabla \mathcal{M}_1(x; \pi, \rho), \tag{6.29c}$$

$$\|Ax - b\| \leq \eta_*, \tag{6.29d}$$

$$x \cdot z \leq \omega_* e, \tag{6.29e}$$

where η_* and ω_* are the feasibility and optimality tolerances, respectively.

Algorithm 6.3: Bound-constrained Lagrangian algorithm

Set initial penalty parameters: $\mu_0 < 1$, $\tau < 1$;

Choose convergence tolerances $\omega_*, \eta_* \ll 1$;

Set constants $\alpha, \beta > 0$ with $\alpha < 1$;

`converged` $\leftarrow$ **false**;

Let $\pi_0 = \pi_e$;

while **not** `converged` **do**

  Choose $\omega_k \geq \omega_*$ such that $\lim_{k\to\infty} \omega_k = \omega_*$.

  Solve (6.16) to obtain solution (x_k^*, z_k^*) satisfying (6.28);

  if $\|Ax_k^* - b\| \leq \max\{\eta_k, \eta_*\}$ **then**

    $x_{k+1} = x_k^*; \quad z_{k+1} = z_k^*; \quad \pi_{k+1} = \pi_k - \frac{1}{\mu_k}(Ax_k^* - b)$;

    if $(x_{k+1}, \pi_{k+1}, z_{k+1})$ satisfies condition (6.29) **then**

      `converged` $\leftarrow$ **true**;

    end if

    $\mu_{k+1} = \mu_k$; [keep μ_k]

    $\eta_{k+1} = \mu_{k+1}^{\beta}\eta_k$ [decrease η_k]

  else

    $\mu_{k+1} = \tau\mu_k$; [decrease μ_k]

    $x_{k+1} = x_k; \quad z_{k+1} = z_k; \quad \pi_{k+1} = \pi_k$;

    $\eta_{k+1} = \mu_{k+1}^{\alpha}\eta_0$ [increase or decrease η_k]

  end if

  $k \leftarrow k + 1$;

end while

---

7 Solving the KKT Systems

At each iteration of the quadratic programming methods, it is necessary to solve one or two KKT systems. In this chapter, two alternative approaches for solving the systems are considered. The first approach involves the symmetric transformation of the reduced Hessian matrix. The second approach uses a symmetric indefinite factorization of a fixed KKT matrix in conjunction with the factorization of a smaller matrix that is updated at each iteration.

7.1 Variable-Reduction Method

The variable-reduction method involves transforming a KKT equation to block-triangular form using a nonsingular block-diagonal matrix. Instead of solving the reduced KKT system normally associated with the standard-form algorithm, the variable-reduction method focuses on solving the full KKT system. Therefore, in this section, we consider a generic full KKT system of the form

$$\begin{pmatrix} H & A^T & P_N \\ A & & \\ P_N^T & & \end{pmatrix} \begin{pmatrix} y \\ w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} h \\ f_1 \\ f_2 \end{pmatrix}. \tag{7.1}$$

First consider a column permutation P such that $AP = \begin{pmatrix} B & S & N \end{pmatrix}$, with B an $m \times m$ nonsingular matrix and S an $m \times n_S$ matrix with $n_S = n_B - m$. The matrix P is a version of the permutation $\begin{pmatrix} P_B & P_N \end{pmatrix}$ of (3.2) that also arranges the columns of A_B in the form $A_B = \begin{pmatrix} B & S \end{pmatrix}$. The n_S variables associated with S are called the *superbasic* variables. Given P, consider the nonsingular $n \times n$ matrix Q such that

$$Q = P \begin{pmatrix} -B^{-1}S & I_m & 0 \\ I_{n_S} & 0 & 0 \\ 0 & 0 & I_N \end{pmatrix}.$$

The columns of Q may be partitioned so that $Q = \begin{pmatrix} Z & Y & W \end{pmatrix}$, where

$$Z = P\begin{pmatrix} -B^{-1}S \\ I_{n_S} \\ 0 \end{pmatrix}, \quad Y = P\begin{pmatrix} I_m \\ 0 \\ 0 \end{pmatrix} \quad \text{and} \quad W = P\begin{pmatrix} 0 \\ 0 \\ I_N \end{pmatrix}.$$

The columns of the $n \times n_S$ matrix Z form a basis for the null-space of A_w with

$$A_w Q = \begin{pmatrix} A \\ P_N^T \end{pmatrix} Q = \begin{pmatrix} 0 & B & N \\ 0 & 0 & I_N \end{pmatrix}.$$

Multiplying the KKT matrix in (7.1) by the diagonal-block matrix $\mathrm{diag}(Q, I_m)$ leads to

$$\begin{pmatrix} Z^THZ & Z^THY & Z^THW & & \\ Y^THZ & Y^THY & Y^THW & B^T & \\ W^THZ & W^THY & W^THW & N^T & I_N \\ & B & N & & \\ & & I_N & & \end{pmatrix}\begin{pmatrix} y_Z \\ y_Y \\ y_W \\ w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} h_Z \\ h_Y \\ h_W \\ f_1 \\ f_2 \end{pmatrix}, \tag{7.2}$$

with $h_Z = Z^Th$, $h_Y = Y^Th$, and $h_W = W^Th$. Then the vector y may be computed as $y = Yy_Y + Zy_Z + Wy_W$. Additionally,

$$\begin{aligned} y_W &= f_2, & & & & \\ By_Y &= f_1 - Nf_2, & y_R &= Yy_Y + Wy_W, & & \\ Z^THZy_Z &= Z^T(h - Hy_R), & y_T &= Zy_Z, & y &= y_R + y_T, \\ B^Tw_1 &= Y^T(h - Hy), & w_2 &= W^T(h - Hy) - N^Tw_1. & & \end{aligned}$$

These equations may be solved using a Cholesky factorization of Z^THZ and an LU factorization of B. The factors of B allow efficient calculation of matrix-vector products Z^Tv or Zv without the need to form the inverse of B.

7.1.1 Equations for the standard-form algorithm

The equations simply considerably when the appropriate right-hand sides from the standard-form nonbinding-direction method are substituted into the above equations. For System 1 (3.3), the substitutions are

$$y = p, \quad w_1 = -q_\pi, \quad w_2 = -q_N, \quad h = f_1 = 0, \quad \text{and} \quad f_2 = e_s,$$

leading to the equations

$$\begin{aligned}
Bp_Y &= -a_{\nu_s}, & p_R &= P\begin{pmatrix} p_Y \\ 0 \\ e_s \end{pmatrix}, & & \\
Z^T H Z p_Z &= -Z^T H p_R, & p_T &= Z p_Z, & p &= p_R + p_T, \\
B^T q_\pi &= (Hp)_{\mathcal{BB}}, & q_N &= (Hp - A^T q_\pi)_{\mathcal{N}}, & q &= \begin{pmatrix} q_\pi \\ q_N \end{pmatrix}.
\end{aligned} \tag{7.3}$$

Similarly for System 2, it holds that $u_W = 0$, $u_Y = 0$, $u_R = 0$ and

$$\begin{aligned}
Z^T H Z u_Z &= Z^T e_{\beta_r}, & u &= Z u_Z, & & \\
B^T v_\pi &= (Hu - e_{\beta_r})_{\mathcal{BB}}, & v_N &= (Hu - A^T v_\pi)_{\mathcal{N}}, & v &= \begin{pmatrix} v_\pi \\ v_N \end{pmatrix}.
\end{aligned} \tag{7.4}$$

The subscript $\mathcal{BB}$ refers to the indices forming B in A_B (a subset of the basic set $\mathcal{B}$). These equations allow us to specialize part 2(a) of Result 3.2.2, which gives the conditions for the linear independence of the matrix A_B.

Result 7.1.1. *Let x be a subspace minimizer with respect to $\mathcal{B}$. Assume that p and q are defined by* (7.3)*, and that x_{β_r} is the incoming nonbasic variable at the next iterate. Let vectors u_B and v_π be defined by* (7.4)*.*

(a) *If x_{β_r} is superbasic, then e_r and the rows of A_B are linearly independent.*

(b) *If x_{β_r} is not superbasic, then e_r and the rows of A_B are linearly independent if and only if $S^T z \neq 0$, where z is the solution of $B^T z = e_r$.*

Proof. From (7.4), $u = Zu_Z$, which implies that u_B is nonzero if and only if u_Z is nonzero. Similarly, the nonsingularity of $Z^T H Z$ implies that u_Z is nonzero if and only if $Z^T e_{\beta_r}$ is nonzero. Now

$$Z^T e_{\beta_r} = \begin{pmatrix} -S^T B^{-T} & I_{n_S} & 0 \end{pmatrix} e_r.$$

If $r > m$, then x_{β_r} will change from being superbasic to nonbasic, and $Z^T e_r = e_{r-m} \neq 0$. However, if $r \leq m$, then

$$Z^T e_{\beta_r} = -S^T B^{-T} e_r = -S^T z,$$

where z is the solution of $B^T z = e_r$. $\square$

Variable-reduction is most efficient when the size of the reduced Hessian ($n_S = n - m - n_N$) is small, i.e., when many constraints are active. This method is used in the current versions of SQOPT [39] and SNOPT [38].

7.2 Schur-Complement and Block-LU Method

In this section, we consider a method for solving the reduced KKT system of the form

$$\begin{pmatrix} H_B & A_B^T \\ A_B & -\mu I \end{pmatrix} \begin{pmatrix} y \\ w \end{pmatrix} = \begin{pmatrix} h \\ f \end{pmatrix}, \tag{7.5}$$

where h and f are constant vectors defined by the algorithm implemented.

Solving a single linear system can be done very effectively using sparse matrix factorization techniques. However, within a QP algorithm, many closely related systems must be solved where the KKT matrix differs by a single row and column. Instead of reformulating the matrix at each iteration, the matrix may be "bordered" in a way that reflects the changes to the basic and nonbasic sets (see Bisschop and Meeraus [4], and Gill et al. [42]).

7.2.1 Augmenting the KKT matrix

Let $\mathcal{B}_0$ and $\mathcal{N}_0$ denote the initial basic and nonbasic sets that define the KKT system in (7.5). There are four cases to consider:

(1) a nonbasic variable moves to the basic set and is not in $\mathcal{B}_0$,

(2) a basic variable in $\mathcal{B}_0$ becomes nonbasic,

(3) a basic variable not in $\mathcal{B}_0$ becomes nonbasic, and

(4) a nonbasic variable moves to the basic set and is in $\mathcal{B}_0$.

For case (1), let ν_s be the nonbasic variable that has become basic. The next KKT matrix can be written as

$$\left(\begin{array}{cc|c} H_B & A_B^T & (h_{\nu_s})_{\mathcal{B}_0} \\ A_B & -\mu I & a_{\nu_s} \\ \hline (h_{\nu_s})_{\mathcal{B}_0}^T & a_{\nu_s}^T & h_{\nu_s,\nu_s} \end{array}\right).$$

Suppose that at the next stage, another nonbasic variable ν_r becomes basic. The KKT matrix is augmented in a similar fashion, i.e.,

$$\left(\begin{array}{ccc|c} H_B & A_B^T & (h_{\nu_s})_{\mathcal{B}_0} & (h_{\nu_r})_{\mathcal{B}_0} \\ A_B & -\mu I & a_{\nu_s} & a_{\nu_r} \\ (h_{\nu_s})_{\mathcal{B}_0}^T & a_{\nu_s}^T & h_{\nu_s,\nu_s} & h_{\nu_s,\nu_r} \\ \hline (h_{\nu_r})_{\mathcal{B}_0}^T & a_{\nu_r}^T & h_{\nu_r,\nu_s} & h_{\nu_r,\nu_r} \end{array}\right).$$

Now consider case (2) and let $\beta_r \in \mathcal{B}_0$ become nonbasic. The change to the basic set is reflected in the new KKT matrix

$$\left(\begin{array}{cccc|c} H_B & A_B^T & (h_{\nu_s})_{\mathcal{B}_0} & (h_{\nu_r})_{\mathcal{B}_0} & e_r \\ A_B & -\mu I & a_{\nu_s} & a_{\nu_r} & 0 \\ (h_{\nu_s})_{\mathcal{B}_0}^T & a_{\nu_s}^T & h_{\nu_s,\nu_s} & h_{\nu_s,\nu_r} & 0 \\ (h_{\nu_r})_{\mathcal{B}_0}^T & a_{\nu_r}^T & h_{\nu_r,\nu_s} & h_{\nu_r,\nu_r} & 0 \\ \hline e_r^T & 0 & 0 & 0 & 0 \end{array}\right).$$

The unit row and column augmenting the matrix has the effect of zeroing out the components corresponding to the removed basic variable.

In case (3), the basic variable must have been added to the basic set at a previous stage as in case (1). Thus, removing it from the basic set can be done by removing the row and column in the augmented part of the KKT matrix corresponding to its addition to the basic set. For example, if ν_s is the basic to be removed, then the new KKT matrix is given by

$$\begin{pmatrix} H_B & A_B^T & (h_{\nu_r})_{\mathcal{B}_0} & e_r \\ A_B & -\mu I & a_{\nu_r} & 0 \\ (h_{\nu_r})_{\mathcal{B}_0}^T & a_{\nu_r}^T & h_{\nu_r,\nu_r} & 0 \\ e_r^T & 0 & 0 & 0 \end{pmatrix}.$$

For case (4), a nonbasic variable in $\mathcal{B}_0$ implies that at some previous stage, the variable was removed from $\mathcal{B}_0$ as in case (2). The new KKT matrix can be formed by removing the unit row and column in the augmented part of the KKT matrix corresponding to the removal the variable from the basic set. In this example, the new KKT matrix becomes

$$\begin{pmatrix} H_B & A_B^T & (h_{\nu_r})_{\mathcal{B}_0} \\ A_B & -\mu I & a_{\nu_r} \\ (h_{\nu_r})_{\mathcal{B}_0}^T & a_{\nu_r}^T & h_{\nu_r,\nu_r} \end{pmatrix}.$$

After k iterations, the KKT system is maintained as a symmetric augmented system of the form

$$\begin{pmatrix} K & V \\ V^T & B \end{pmatrix} \begin{pmatrix} r \\ \eta \end{pmatrix} = \begin{pmatrix} b \\ f \end{pmatrix} \quad \text{with} \quad K = \begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix}, \tag{7.6}$$

where B is of dimension at most $2k$.

7.2.2 Factoring the matrices

Although the augmented system (in general) increases in dimension by one at every iteration, the $(1,1)$–block K is fixed and defined by the initial set of basic variables. The *Schur* *complement method* assumes that factorizations for K and the *Schur complement* $C = B - V^T K^{-1} V$ exist. Then the solution of (7.6) can be determined by solving the equations

$$Kt = b, \quad C\eta = f - V^T t, \quad Kr = b - V\eta.$$

The work required is dominated by two solves with the fixed matrix K and one solve with the Schur complement C. If the number of changes to the basic set is small enough, dense factors of C may be maintained.

The Schur complement method can be extended to a *block-LU method* by storing the augmented matrix in block factors such that

$$\begin{pmatrix} K & V \\ V^T & B \end{pmatrix} = \begin{pmatrix} L & \\ Z^T & I \end{pmatrix} \begin{pmatrix} U & Y \\ & C \end{pmatrix}, \tag{7.7}$$

where $K = LU$, $LY = V$, $U^T Z = V$, and $C = B - Z^T Y$ is the Schur-complement matrix.

The solution of (7.6) with factors (7.7) can be computed by forming the block factors and by solving the equations

$$Lt = b, \quad C\eta = f - Z^T t, \ \text{ and } \ Ur = t - Y\eta.$$

This method requires a solve with L and U each, one multiply with Y and Z^T, and one solve with the Schur complement C. For more details, see Gill et al. [40], Eldersveld and Saunders [27], and Huynh [53].

As the iterations of the QP algorithm proceed, the size of C increases and the work required to solve with C increases. It may be necessary to restart the process by discarding the existing factors and re-forming K based on the current set of basic variables.

Using the LDL^T factorization Since K is a symmetric indefinite matrix, K can be factored using an LDL^T factorization rather than an LU factorization (see Result 1.3.4). Given such a factorization, the augmented matrix can be stored in the form

$$\begin{pmatrix} K & V \\ V^T & B \end{pmatrix} = \begin{pmatrix} L & \\ Y^T & I \end{pmatrix} \begin{pmatrix} D & \\ & C \end{pmatrix} \begin{pmatrix} L^T & Y \\ & I \end{pmatrix}, \tag{7.8}$$

In this case, less storage is required because only the LDL^T factors, the Schur complement and Y are stored. The solution of (7.6) is computed from the equations

$$Lt = b, \quad C\eta = f - Y^T t, \quad Dy = t, \ \text{ and } \ L^T r = v - Y\eta,$$

requiring a solve with each of the matrices L, D, L^T and C and a multiply with Y and its transpose.

7.2.3 Updating the factors

Suppose the current KKT matrix is bordered by the vectors v and w, and the scalar σ

$$\left(\begin{array}{cc|c} K & V & v \\ V^T & B & w \\ \hline v^T & w^T & \sigma \end{array}\right).$$

The block-LU factors Y and Z, and the Schur complement C in (7.7) are updated every time the system is bordered. The number of columns in matrices Y and Z and the dimension of the Schur complement increase by one. The updates y, z, c and γ are defined by the equations

$$\begin{aligned} Ly &= v, & U^T z &= v, \\ c = w - Z^T y &= w - Y^T z, & \gamma &= \sigma - z^T y, \end{aligned}$$

so that the new block-LU factors satisfy

$$\left(\begin{array}{cc|c} K & V & v \\ V^T & B & w \\ \hline v^T & w^T & \sigma \end{array}\right) = \left(\begin{array}{c|cc} L & & \\ Z^T & I & \\ \hline z^T & & 1 \end{array}\right) \left(\begin{array}{c|cc} U & Y & y \\ & C & c \\ \hline & c^T & \gamma \end{array}\right). \tag{7.9}$$

As demonstrated previously, it is also possible to border the KKT matrix with two rows and columns in one iteration (e.g., a swap involving the removal of an original basic variable (case (2) in Section 7.2.1) and the addition of a new nonbasic variable (case (1)). The above updates still apply but with appropriate expansions of the vectors and scalars in the equations.

8 Numerical Results

In this chapter, numerical results are presented for the Fortran package icQP [45], which is an implementation of the nonbinding-direction method for QPs in standard-form. The results are compared with those of the convex QP solver SQOPT [39].

Problems were taken from the CUTEr problem collection [6, 50], and the Maros and Mészáros convex quadratic programming set [56]. A total of 171 quadratic problems in the CUTEr set were identified based on the classification code, while 138 convex quadratic programs were taken from the Maros and Mészáros test set. Only 126 of the 171 CUTEr problems were included in the icQP test set. 45 of the problems (those with names prefixed by `A0`, `A2` and `A5`) were deemed too expensive to include. In over twelve hours, icQP solved only 13 of the 45 problems. In the Maros and Mészáros set, problems `BOYD1` and `BOYD2` were also excluded for the same reason.

The number of constraints `m` and variables `n` for the CUTEr and Maros and Mészáros sets are given in Tables A.1 and A.2. The superscript i denotes a nonconvex problem. The CUTEr problems are written in Standard Input Format (SIF), while the Maros and Mészáros problems are written in QPS format, a subset of the SIF format. The CUTEr testing environment [50], which includes the SIF decoder SifDec, was used to pass the problem data into icQP.

Results were obtained on an iMac with a 2.8 GHz Intel Core i7 processor and 16GB of RAM. All software was compiled using gfortran 4.6.0 with code optimization flag `-O3`.

8.1 Implementation

icQP is a Fortran 2003 implementation of the standard-form version of the nonbinding-direction algorithm presented in Section 3.2. The problem is assumed to be of the form

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; \varphi(x) \quad \text{subject to} \;\; \ell \le \begin{pmatrix} x \\ Ax \end{pmatrix} \le u,$$

where $\varphi(x)$ is a linear or quadratic objective function, ℓ and u are constant lower and upper bounds, and A is an $m \times n$ matrix. The objective function has the form

$$\varphi(x) = \varphi_0 + c^T x + \tfrac{1}{2} x^T H x,$$

where φ_0 is a scalar constant that does not affect the optimization. Internally, the problem is converted to standard-form by introducing slack variables s such that

$$\underset{x\in\mathbb{R}^n, s\in\mathbb{R}^m}{\text{minimize}} \quad \varphi(x) \quad \text{subject to} \quad Ax - s = 0, \quad \ell \le \begin{pmatrix} x \\ s \end{pmatrix} \le u.$$

An initial feasible point is found via a phase 1 LP using an experimental Fortran 90 version of SNOPT. This process also produces an initial basis. If this basis is not second-order-consistent, then the number of non-positive eigenvalues of the KKT matrix is greater than m, and the estimated number of temporary constraints e_a is defined as the difference of these numbers. If the estimate satisfies

$$e_a > \max\{\tfrac{1}{2}(n_B - m), 10\}, \tag{8.1}$$

then a vertex is defined at the current point by temporarily fixing variables at their current values. Otherwise, the method described in Section 5.2.2 is used to define temporary constraints that define a second-order-consistent basis.

Three linear solvers have been incorporated into icQP to store the block-LU (or block-LDL^T) factors of the KKT matrix. These are LUSOL [41], HSL_MA57 [25], and UMFPACK [15, 16, 17, 18]. The Schur complement matrix is maintained by the dense matrix factorization code LUMOD [62]. LUMOD was updated to Fortran 90 by Huynh [53] for the QP code QPBLU, which also utilizes a block-LU scheme. Modifications were made to the Fortran 90 version of LUMOD to incorporate it into icQP.

The algorithm for computing temporary constraints for a second-order-consistent basis requires a solver that computes an LDL^T factorization and provides access to the matrix L. Only HSL_MA57 is a symmetric indefinite solver, but it does not provide access to L by default. However, a subroutine returning L was provided by Iain Duff [26], and so HSL_MA57 is the only solver capable of defining temporary constraints using the method of Section 5.2.2. For all other solvers, a vertex is defined if the initial basis is not second-order-consistent.

Table 8.1 lists the problems for which the phase 1 LP did not provide a second-order-consistent basis when running icQP with HSL_MA57. Based on whether or not (8.1) holds, either variables were temporarily fixed at their current values to create a vertex, or temporary constraints were computed to create an initial second-order-consistent basis. The superscript v denotes the former case. The column labeled "`nTmp`" gives the number of temporary constraints or temporarily fixed variables. Column "`Dense`" gives the density of $H_B + \rho A_B^T A_B$. Column "`Time`" gives the time taken to compute the temporary constraints and factor the resulting KKT matrix. The column "`nFix`" of Tables 8.7 and 8.8 list the number of fixed variables needed to define a temporary vertex.

The condition for a blocking variable to give a linearly dependent basis is $u = 0$, where u satisfies the equations (3.6). The test used in icQP is

$$\|u_B\|_\infty < \tau,$$

Table 8.1: Number of temporary constraints for icQP with HSL_MA57

Name	nTmp	Dense	Time	Name	nTmp	Dense	Time
BLOCKQP1	5007^v	0.00	0.38	NCVXQP2	446^v	0.00	0.10
BLOCKQP2	1	49.97	101.44	NCVXQP3	155	0.54	0.23
BLOCKQP3	5007^v	0.00	0.38	NCVXQP4	731^v	0.00	0.03
BLOCKQP4	1	49.97	101.53	NCVXQP5	731^v	0.00	0.03
BLOCKQP5	5003^v	0.00	0.37	NCVXQP6	221	0.47	0.09
BLOWEYA	1	12.55	27.25	NCVXQP7	199^v	0.00	0.29
BLOWEYB	1	12.55	27.25	NCVXQP8	199^v	0.00	0.33
BLOWEYC	1	12.55	27.26	NCVXQP9	199^v	0.00	0.29
GMNCASE1	1	22.03	0.07	STATIC3	58	0.95	0.00
HATFLDH	1	31.25	0.00	STNQP1	348	0.07	2.32
HS44NEW	3	28.00	0.00	STNQP2	769	0.12	3.84
NCVXQP1	446^v	0.00	0.10				

where τ is a scaled tolerance that is initialized at $\tau = (\max(\|A\|_\infty, \|H\|_\infty) + 1) \times 9 \times 10^{-12}$, and increased, if necessary, subject to the fixed maximum value 5×10^{-7}. The condition for increasing τ is based on the norm of u_B. If $\|u_B\|_\infty$ is close to τ, specifically, if $\|u_B\|_\infty$ satisfies

$$0 < \frac{\|u_B\|_\infty - \tau}{\tau} < 12,$$

then the tolerance is increased from τ to 12τ.

The KKT matrix is refactored when the dimension of the Schur complement becomes greater than $\min(1000, \frac{1}{2}(n_B + m))$, or when the estimated condition number of the Schur complement is greater than 10^8. The maximum dimension of the Schur complement was determined empirically, and was based on the overall time required to solve the problems with large numbers of degrees of freedom (see Figures 8.5–8.6). Ideally, this limit should be chosen to balance the time required to factor the KKT matrix and cumulative time needed to update the Schur complement.

After the KKT matrix is factorized, the current x and π are updated using one step of iterative refinement based on increments p_B and q_π found by solving the additional system

$$\begin{pmatrix} H_B & A_B^T \\ A_B & \end{pmatrix} \begin{pmatrix} p_B \\ -q_\pi \end{pmatrix} = -\begin{pmatrix} g_B - A_B^T \pi \\ 0 \end{pmatrix}.$$

8.2 Performance Profiles

Performance profiles were created to analyze the results of the numerical experiments on icQP. The merits of using performance profiles to benchmark optimization software are discussed in [20]. The idea of a performance profile is to provide an "at-a-glance" comparison of the performance of a set $\mathcal{S}$ of n_s solvers applied to a test set $\mathcal{P}$ of n_p problems. For each solver $s \in \mathcal{S}$

and problem $p \in \mathcal{P}$ in a profile, the number t_{ps} is the time (or some other measure, e.g., number of iterations) needed to solve problem p with solver s. To compare the performance of a problem p over the different solvers, the *performance ratio* for each successfully solved problem and solver is defined as

$$r_{ps} = \frac{t_{ps}}{\min\{t_{ps} : s \in \mathcal{S}\}}.$$

If r_{ms} denotes the maximum time needed over all problems that were solved successfully, then the performance ratio for problems that failed is defined as some value greater than r_{ms}.

Given the set of performance ratios, a function $P_s(\sigma)$ is defined for each solver such that

$$P_s(\sigma) = \frac{1}{n_p}|\{p \in \mathcal{P} : r_{ps} \leq \sigma\}|,$$

where $\sigma \in [1, r_{ms}]$. The value $P_s(\sigma)$ is the fraction of problems for solver s that were solved within σ of the best time. $P_s(1)$ is the fraction of problems for which s was the fastest solver. Note that the summation of $P_s(1)$ for all s does not necessarily equal one, because there may be ties in the times (e.g., a "0" is recorded if a problem is solved in less than 10^{-3} seconds). The value $P_s(r_{ms})$ gives the fraction of problems solved successfully by solver s.

The presented performance profiles are log-scaled, with $\tau = \log_2(\sigma)$ on the x-axis and the function

$$P_s(\tau) = \frac{1}{n_p}|\{p \in \mathcal{P} : \log_2(r_{ps}) \leq \tau\}|,$$

on the y-axis for each solver. The y-axis can be interpreted as the fraction of problems that were solved within 2^τ of the best time. Because the y-axis is the fraction of problems solved, and the x-axis is the factor of time needed to solve a problem, the "best" solver should have a function $P_s(\tau)$ that lies towards the upper-left of the graph.

Performance profiles in this chapter were produced using a MATLAB m-file adapted from one given in [13]. If a problem is solved in 0.00 seconds, then that value is replaced by 0.001 to prevent division by zero in the calculation of the performance ratios.

8.3 Results for the Nonbinding-Direction Method

Results were gathered from running the convex QP package SQOPT and four versions of icQP on the CUTEr and Maros and Mészáros test sets. The versions of icQP are:

(1) icQP with LUSOL,

(2) icQP with HSL_MA57,

(3) icQP with UMFPACK, and

(4) icQP with HSL_MA57 starting at a vertex (referred to as HSL_MA57v).

Each version is referred to as icQP-[solver] in the following sections. It must be emphasized that icQP with LUSOL, UMFPACK and HSL_MA57v start with a vertex, while icQP–HSL_MA57 starts with any basic set that defines a subspace minimizer. In particular, icQP–HSL_MA57 is the only version capable of using temporary constraints to define a second-order-consistent basis (see Section 5.2.2).

Default parameter settings were used throughout, including the third-party linear algebra solvers. The only exception was matrix scaling, which was turned off for all the solvers.

8.3.1 Results on icQP with different linear solvers

In this section, we compare the performance of icQP for each of the linear solvers LUSOL, HSL_MA57, UMFPACK, and HSL_MA57v. The performance of a given solver depends greatly on the Fortran interface to icQP. Each solvers requires a different matrix input format (e.g., in symmetric/unsymmetric coordinate form, or sparse-by-column format), and the timing often depends on the efficiency of the implementation. In the case of HSL_MA57, performance was inhibited by the fact that the solver was not designed to be called repeatedly within an iterative scheme.

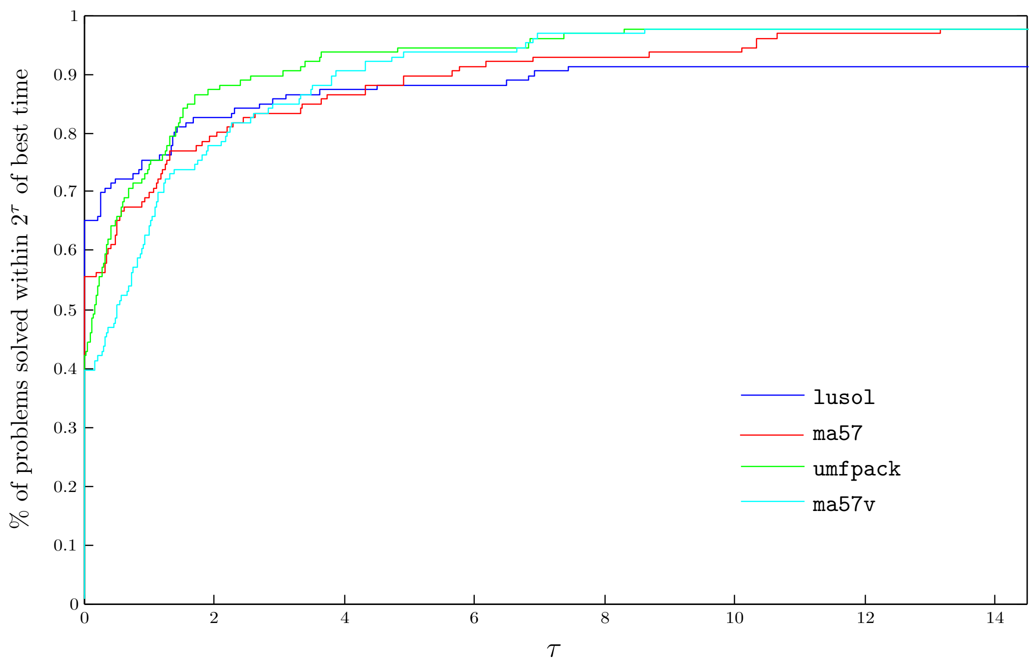

Figure 8.1: Performance profile of solve times for icQP on the CUTEr QP test set.

The performance profile for icQP on the CUTEr test set is given in Figure 8.1. Although icQP–LUSOL solves the most problems in the best time, it solved fewer problems than the other versions of icQP. No version of icQP was able to solve the CUTEr problems `CVXQP1`, `CVXQP3` and `CONT5-QP` in the CUTEr set. In addition, icQP–LUSOL was unable to solve the problems `KSIP`, QPCBLEND, QPCSTAIR, QPNBLEND, and QPNSTAIR.

Broadly speaking, the results on the Maros and Mészáros test set mirrored those for the CUTEr test set, although the times for icQP–UMFPACK had a slight edge over those for icQP–HSL_MA57 and icQP–HSL_MA57v. The performance profile is given in Figure 8.2.

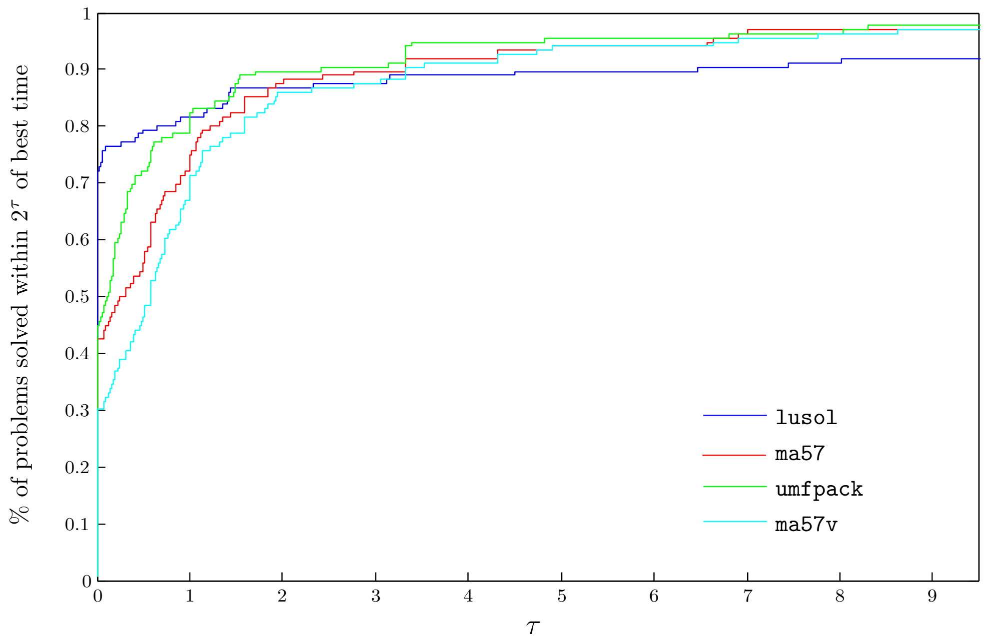

Figure 8.2: Performance profile of solve times for icQP on the Maros and Mészáros QP test set.

For the Maros and Mészáros set, no version of icQP was able to solve QPILOTNO, CVXQP1_L, and CVXQP3_L. The versions icQP–HSL_MA57 and icQP–HSL_MA57v also failed to solve Q25FV47. icQP–LUSOL failed to solve CVXQP2_L, HUESTIS, KSIP, MOSARQP1, MOSARQP2, Q25FV47, QPCBLEND, QPILOTNO, and UBH1. The larger number of failures in the Maros and Mészáros set may be caused by the limitations of QPS format, which specifies only 12 characters for a numeric field, thus limiting the precision of the data. In fact, on inspection of the QPS files, many problems had far fewer than 12 digits of precision.

Table 8.2 illustrates the potential benefit of icQP–HSL_MA57, which uses the method of Section 5.2.2 to define an initial second-order consistent basis. Observe that icQP–HSL_MA57v, which is forced start at a temporary vertex, requires substantially more iterations in all cases. The improvement is most evident in problems AUG2DC, AUG3DC, and GRIDNETB, which are started at an optimal solution and therefore require no iterations in icQP–HSL_MA57.

8.3.2 Results on icQP and SQOPT

Since SQOPT is a convex QP solver, only convex problems were chosen for the comparison with icQP. Nonconvex problems are denoted by a superscript i in Table A.1. The 90 convex

Table 8.2: Results on a subset of problems from the CUTEr set for icQP–HSL_MA57 and icQP–HSL_MA57v

	icQP-HSL_MA57			icQP-HSL_MA57v		
Name	Objective	Itn	Time	Objective	Itn	Time
AUG2D	1.6874E+06	396	3.28	1.6874E+06	10193	86.56
AUG2DC	1.8184E+06	1	0.86	1.8184E+06	10201	86.58
AUG3DC	2.7654E+04	1	1.78	2.7654E+04	19544	224.60
DTOC3	2.3526E+02	3	0.48	2.3526E+02	4806	189.00
GRIDNETA	3.0498E+02	224	1.03	3.0498E+02	2255	11.58
GRIDNETB	1.4332E+02	1	0.39	1.4332E+02	6561	43.13
HUES-MOD	3.4824E+07	559	1.74	3.4830E+07	9304	24.53
HUESTIS	3.4824E+11	559	1.74	3.4830E+11	9304	24.50

CUTEr problems were divided into two sets. The first set contains 35 problems with number of degrees of freedom (or number of superbasic variables), denoted by `nS`, greater than 1000 or $\frac{1}{2}(m+n)$. The second set contains the remaining 55 problems. The Maros and Mészáros set contains only convex problems, so all of those problems were included in the comparison. The small/large `nS` partition of Maros and Mészáros problems resulted in a "small `nS`" set of 115 problems and a "large `nS`" set with 21 problems.

SQOPT. SQOPT uses a reduced-Hessian active-set method implemented as a reduced-gradient method. The solver partitions the equality constraints $Ax - s = 0$ into the form $Bx_B + Sx_S + Nx_N = 0$, where the *basis matrix* B is nonsingular and $m \times m$, and S and N are the remaining columns of the matrix $\begin{pmatrix} A & -I \end{pmatrix}$. The vectors x_B, x_S and x_N are the basic, superbasic, and nonbasic components of (x, s). Given this partition, a matrix Z with columns spanning the null space of the active constraints can be defined as

$$Z = P \begin{pmatrix} -B^{-1}S \\ I \\ 0 \end{pmatrix},$$

where P is the permutation matrix that permutes $\begin{pmatrix} A & -I \end{pmatrix}$ into $\begin{pmatrix} B & S & N \end{pmatrix}$ (for more details, see Section 7.1). A suitable direction is computed from an equation involving the reduced Hessian and reduced gradient

$$Z^T H Z p_S = -Z^T g, \tag{8.2}$$

a system with n_S equations. If the number of superbasics is large, then solving (8.2) becomes expensive. By default, SQOPT switches to a conjugate-gradient method to solve for a direction, when n_S is greater than 2000. Therefore, it is expected that SQOPT will provide superior performance when there are few superbasics.

Tables 8.3 and 8.4 list the results for SQOPT and the different versions of icQP on the CUTEr and Maros and Mészáros problems. The column "`Objective`" gives the final objective value, column “`Itn`” is the total number of iterations, and the column “`Time`” lists the total amount of time in seconds. Superscripts on the objective value denote an exit condition. If no superscript is present, then the problem was solved to optimality. Otherwise, a “1” indicates that a problem was declared to be unbounded, “2” implies that a problem was declared to be infeasible, a “i” implies that the problem was declared to be nonconvex, and a “n” indicates that an algorithm exceeded its iteration limit. The superscript “f” indicates that difficulties were encountered when factorizing a KKT matrix; either the matrix was deemed to be singular by the linear solver, or the matrix had incorrect inertia. Failures of this kind were usually caused by poor scaling. Tables 8.5 and 8.6 give the final number of superbasics and the total number of factorizations of the KKT matrix needed for each problem.

Analysis. On CUTEr problems with a small value of `nS`, as expected, SQOPT performed significantly better than every version of icQP. SQOPT has the fastest solve time for over 95% of the problems in this set. The performance profile of the solve times is given in Figure 8.3. Similar performance was observed for the Maros and Mészáros problems with a small value of `nS`. However, SQOPT failed to solve 27 of the 115 problems, while the worst version of icQP was unable to solve 6. This behavior could, again, be attributed to the limitations of QPS format, and also to the lack of scaling in the solvers.

The performance of icQP relative to SQOPT improves for problems with a large number of superbasics. The performance profile for the 35 convex CUTEr problems with large `nS` is given in Figure 8.5. For this set, icQP–HSL_MA57 appears to have the best performance, with 60% of the best times. No version of icQP was able to solve `CVXQP1`. In addition, icQP–LUSOL was unable to solve `HUESTIS`, `MOSARQP1`, and `UBH1`. SQOPT failed to solve only `UBH1`.

The icQP's improvement is more dramatic on the Maros and Mészáros set, with the profile of icQP–HSL_MA57 residing securely in the top-left corner of the graph in Figure 8.6. icQP–HSL_MA57 gives the best time on the same number of problems as SQOPT, but also solves most of the problems within a factor of 12 of the best time.

These results are combined in the performance profile in Figure 8.7, which graphs the performance of SQOPT and icQP–HSL_MA57 on the convex CUTEr and Maros and Mészáros problems with a large number of superbasics. SQOPT is more robust, but icQP–HSL_MA57 solves almost 70% of the problems in a faster time.

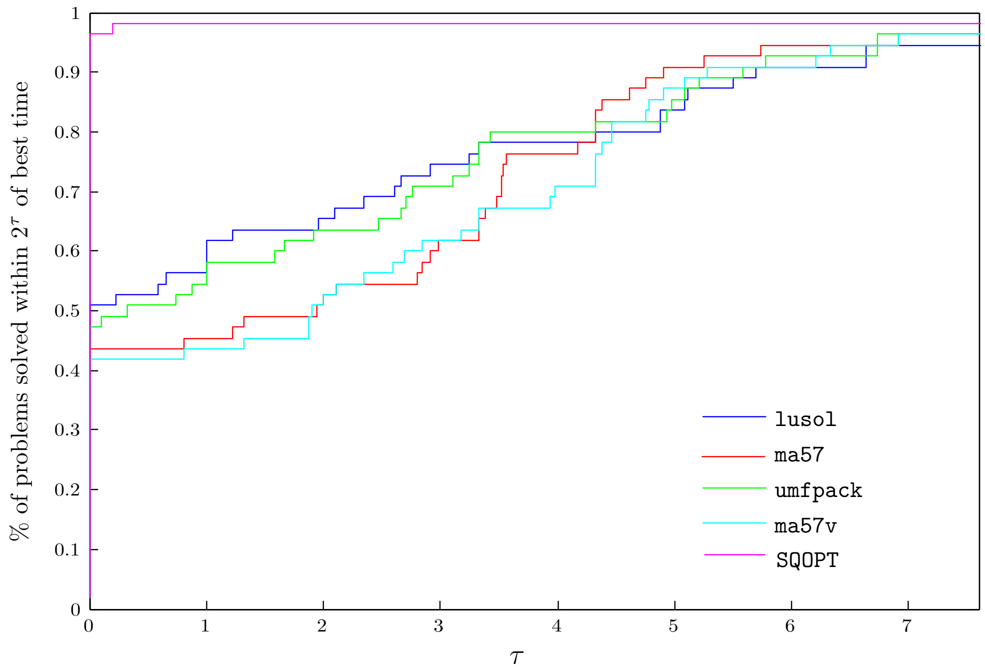

Figure 8.3: Performance profile of solve times for SQOPT and icQP on 55 convex CUTEr QPs with a small number of degrees of freedom.

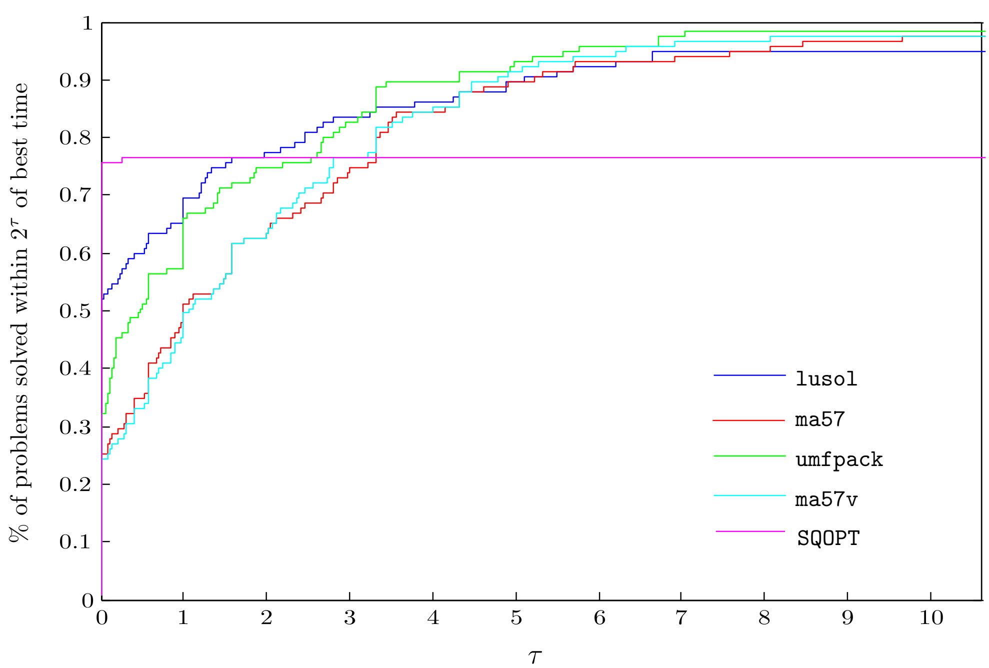

Figure 8.4: Performance profile of solve times for SQOPT and icQP on 115 Maros and Mészáros QPs with a small number of degrees of freedom.

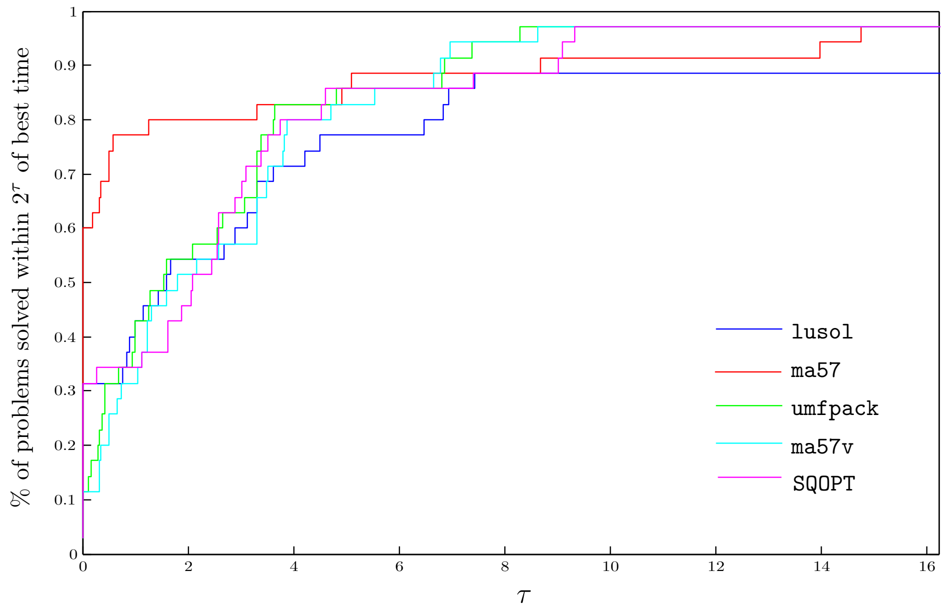

Figure 8.5: Performance profile of solve times for SQOPT and icQP on 35 convex CUTEr QPs with a large number of degrees of freedom.

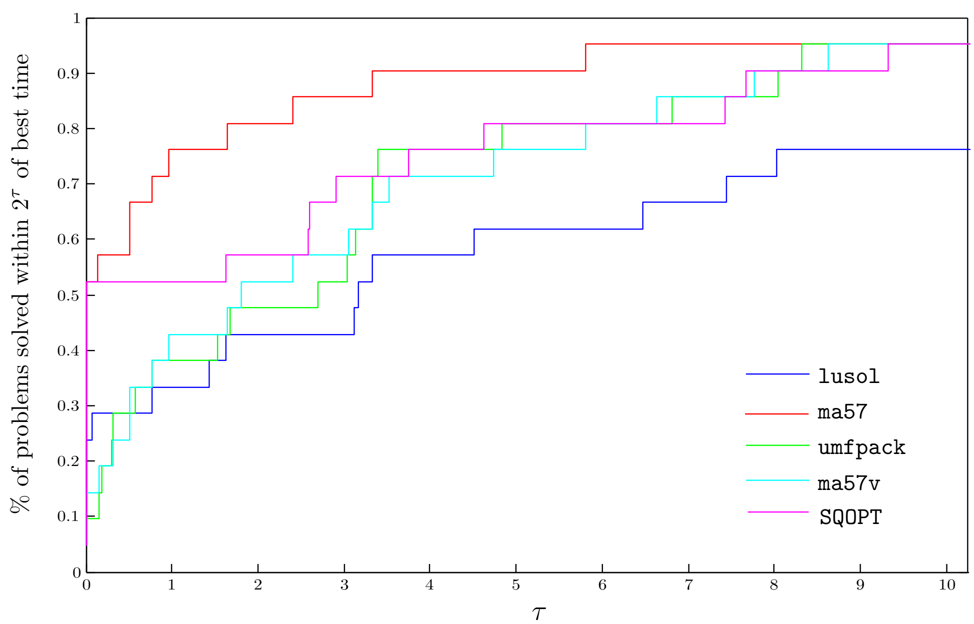

Figure 8.6: Performance profile of solve times for SQOPT and icQP on 21 Maros and Mészáros QPs with a large number of degrees of freedom.

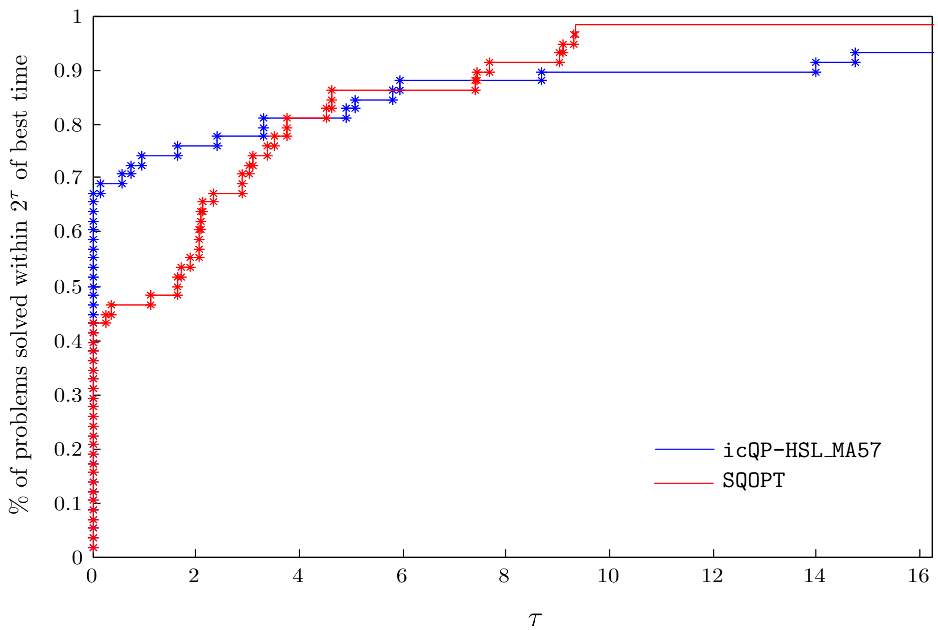

Figure 8.7: Performance profile of solve times for SQOPT and icQP–HSL_MA57 on 56 convex CUTEr and Maros and Mészáros QPs with a large number of degrees of freedom.

Table 8.3: Results for CUTEr QPs

	LUSOL			MA57			UMFPACK			MA57v			SQOPT		
Name	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time
ALLINQP	-1.0945E+03	4816	9.02	-1.0945E+03	3387	10.31	-1.0945E+03	4816	9.76	-1.0945E+03	4816	15.14	-1.0945E+03	4816	52.68
AUG2D	1.6874E+06	10193	74.18	1.6874E+06	396	3.28	1.6874E+06	10193	92.71	1.6874E+06	10193	86.56	1.6874E+06	10615	558.12
AUG2DC	1.8184E+06	10201	77.18	1.8184E+06	1	0.86	1.8184E+06	10201	97.48	1.8184E+06	10201	86.58	1.8184E+06	10622	555.09
AUG2DCQP	6.4981E+06	14479	94.52	6.4981E+06	14334	133.40	6.4981E+06	14361	117.58	6.4981E+06	14334	133.42	6.4981E+06	14472	567.38
AUG2DQP	6.2370E+06	14599	93.65	6.2370E+06	14591	133.18	6.2370E+06	14266	114.37	6.2370E+06	14591	133.22	6.2370E+06	14185	559.78
AUG3D	2.4561E+04	16910	149.77	2.4561E+04	2164	46.90	2.4561E+04	16910	199.59	2.4561E+04	16910	211.61	2.4561E+04	17647	1085.72
AUG3DC	2.7654E+04	19544	217.50	2.7654E+04	1	1.78	2.7654E+04	19544	295.45	2.7654E+04	19544	224.60	2.7654E+04	19707	931.68
AUG3DCQP	6.1560E+04	22187	188.41	6.1560E+04	22201	234.66	6.1560E+04	22177	253.42	6.1560E+04	22201	234.35	6.1560E+04	22466	1027.97
AUG3DQP	5.4229E+04	18510	138.97	5.4229E+04	18455	178.53	5.4229E+04	18505	186.10	5.4229E+04	18455	178.50	5.4229E+04	18287	588.28
AVGASA	-4.6319E+00	9	0.00	-4.6319E+00	8	0.00	-4.6319E+00	9	0.00	-4.6319E+00	9	0.00	-4.6319E+00	8	0.00
AVGASB	-4.4832E+00	9	0.00	-4.4832E+00	8	0.00	-4.4832E+00	9	0.00	-4.4832E+00	9	0.00	-4.4832E+00	8	0.00
BIGGSC4	-2.4375E+01	11	0.00	-2.4375E+01	11	0.00	-2.4375E+01	11	0.00	-2.4375E+01	11	0.00	-3.0000E+00[i]	5	0.00
BLOCKQP1	-4.9940E+03	5014	4.79	-4.9940E+03	5014	6.25	-4.9940E+03	5014	4.95	-4.9940E+03	5014	5.98	-1.2436E+03[i]	12	0.17
BLOCKQP2	-4.9928E+03	7515	328.89	-4.9938E+03	5006	129.18	-4.9928E+03	7515	369.34	-4.9928E+03	7515	150.92	-2.6179E+03[i]	5030	85.05
BLOCKQP3	-2.4950E+03	5014	4.80	-2.4950E+03	5014	6.44	-2.4950E+03	5014	4.97	-2.4950E+03	5014	6.17	-6.1821E+02[i]	9	0.17
BLOCKQP4	-2.4933E+03	8492	723.82	-2.4958E+03	7401	149.39	-2.4933E+03	8492	371.20	-2.4933E+03	8492	167.14	-1.3433E+03[i]	5809	56.56
BLOCKQP5	-2.4950E+03	5020	4.79	-2.4950E+03	5020	6.14	-2.4950E+03	5020	4.94	-2.4950E+03	5020	5.87	-6.1988E+02[i]	13	0.11
BLOWEYA	-2.0050E-05	3	0.04	-2.2781E-02	806	52.03	-2.0050E-05	3	0.13	-2.0050E-05	3	0.04	-2.0050E-05	1	0.01
BLOWEYB	-1.9870E-10	3	0.04	-1.5226E-02	406	44.29	-1.9874E-10	3	0.12	-1.9875E-10	3	0.04	3.0938E-16	1	0.01
BLOWEYC	-8.0100E-05	3	0.04	-1.5246E-02	806	52.04	-8.0100E-05	3	0.13	-8.0100E-05	3	0.04	-8.0100E-05	1	0.01
CONT5-QP	3.2823E+00[2]	7	2.91	6.6505E-02[f]	9	13.28	3.5066E+30[2]	7	2.94	1.4467E+03[2]	11	27.38	6.3630E-03	1087	494.25
CVXQP1	3.5053E+08[2]	2370	2.12	1.6702E+08[f]	6064	809.07	2.4444E+08[2]	2996	679.57	3.5053E+08[f]	2373	29.35	1.0870E+08	12530	16.28
CVXQP2	8.1842E+07	8347	416.03	8.1842E+07	6144	762.70	8.1842E+07	8335	140.47	8.1842E+07	8367	1051.15	8.1842E+07	8784	22.40
CVXQP3	2.9573E+08[2]	5938	29.74	0.0000E+00[f]	6186	248.18	8.1168E+26[2]	5943	50.75	2.9573E+08[f]	5941	180.25	1.1571E+08	10478	12.61
DEGENQP	0.0000E+00	12	0.06	2.8866E-14	12	0.07	0.0000E+00	12	0.05	2.8866E-14	12	0.07	0.0000E+00	11	0.04
DTOC3	2.3526E+02	4806	83.75	2.3526E+02	3	0.48	2.3526E+02	4806	151.84	2.3526E+02	4806	189.00	2.3526E+02	4805	11.79
DUAL1	3.5013E-02	89	0.00	3.5013E-02	89	0.00	3.5013E-02	89	0.01	3.5013E-02	89	0.01	3.5013E-02	82	0.00

1 = problem declared unbounded, 2 = problem declared infeasible, i = problem declared indefinite, f = failed, n = hit iteration limit

Table 8.3: Results for CUTEr QPs (continued)

	LUSOL			MA57			UMFPACK			MA57v			SQOPT		
Name	Objective	nItn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time
DUAL2	3.3734E-02	100	0.01	3.3734E-02	100	0.01	3.3734E-02	100	0.01	3.3734E-02	100	0.01	3.3734E-02	99	0.00
DUAL3	1.3576E-01	107	0.01	1.3576E-01	107	0.01	1.3576E-01	107	0.01	1.3576E-01	107	0.01	1.3576E-01	118	0.01
DUAL4	7.4609E-01	62	0.00	7.4609E-01	62	0.00	7.4609E-01	62	0.00	7.4609E-01	62	0.00	7.4609E-01	67	0.00
DUALC1	6.1553E+03	10	0.00	6.1553E+03	10	0.02	6.1553E+03	10	0.00	6.1553E+03	10	0.02	6.1553E+03	9	0.00
DUALC2	3.5513E+03	5	0.00	3.5513E+03	5	0.02	3.5513E+03	5	0.00	3.5513E+03	5	0.02	3.5513E+03	4	0.00
DUALC5	4.2723E+02	8	0.00	4.2723E+02	8	0.03	4.2723E+02	8	0.00	4.2723E+02	8	0.03	4.2723E+02	7	0.00
DUALC8	1.8309E+04	7	0.00	1.8309E+04	7	0.12	1.8309E+04	7	0.00	1.8309E+04	7	0.12	1.8309E+04	8	0.00
FERRISDC	0.0000E+00	1	0.06	-3.3890E-05	466	3.61	0.0000E+00	1	0.05	4.8219E-27	1	0.08	0.0000E+00	0	0.03
GENHS28	9.2717E-01	3	0.00	9.2717E-01	1	0.00	9.2717E-01	3	0.00	9.2717E-01	3	0.00	9.2717E-01	3	0.00
GMNCASE1	2.6697E-01	96	0.04	2.6697E-01	54	0.10	2.6697E-01	96	0.04	2.6697E-01	96	0.08	2.6697E-01	102	0.02
GMNCASE2	-9.9444E-01	99	0.06	-9.9444E-01	56	0.07	-9.9444E-01	99	0.05	-9.9444E-01	99	0.11	-9.9444E-01	97	0.03
GMNCASE3	1.5251E+00	128	0.07	1.5251E+00	98	0.21	1.5251E+00	128	0.06	1.5251E+00	128	0.11	1.5251E+00	126	0.03
GMNCASE4	5.9469E+03	173	0.12	5.9469E+03	173	0.26	5.9469E+03	173	0.12	5.9469E+03	173	0.26	5.9469E+03	172	0.06
GOULDQP2	1.8512E-12	1	0.34	1.8512E-12	1	0.27	1.8512E-12	1	0.34	1.8512E-12	1	0.27	1.8512E-12	0	0.01
GOULDQP3	2.3796E-05	5725	27.23	2.3796E-05	5725	65.37	2.3796E-05	5725	35.25	2.3796E-05	5725	64.04	2.3796E-05	7511	83.43
GRIDNETA	3.0498E+02	2271	7.72	3.0498E+02	224	1.03	3.0498E+02	2271	8.66	3.0498E+02	2255	11.58	3.0498E+02	2245	10.76
GRIDNETB	1.4332E+02	6561	44.40	1.4332E+02	1	0.39	1.4332E+02	6561	45.41	1.4332E+02	6561	43.13	1.4332E+02	6741	214.96
GRIDNETC	1.4832E+02	5265	30.00	1.4832E+02	2504	13.37	1.4832E+02	5257	32.54	1.4832E+02	5259	31.61	1.4832E+02	5342	153.33
HATFLDH	-2.4500E+01	4	0.00	-2.4500E+01	6	0.00	-2.4500E+01	4	0.00	-2.4500E+01	4	0.00	-2.4500E+01	3	0.00
HS118	6.6482E+02	22	0.00	6.6482E+02	16	0.00	6.6482E+02	22	0.00	6.6482E+02	22	0.00	6.6482E+02	21	0.00
HS21	-9.9960E+01	2	0.00	-9.9960E+01	1	0.00	-9.9960E+01	2	0.00	-9.9960E+01	2	0.00	-9.9960E+01	1	0.00
HS268	-3.6380E-12	7	0.00	-3.6380E-12	3	0.00	3.6380E-12	6	0.00	-1.0914E-11	6	0.00	0.0000E+00	8	0.00
HS35	1.1111E-01	5	0.00	1.1111E-01	2	0.00	1.1111E-01	5	0.00	1.1111E-01	5	0.00	1.1111E-01	5	0.00
HS35I	1.1111E-01	5	0.00	1.1111E-01	2	0.00	1.1111E-01	5	0.00	1.1111E-01	5	0.00	1.1111E-01	5	0.00
HS35MOD	2.5000E-01	2	0.00	2.5000E-01	1	0.00	2.5000E-01	2	0.00	2.5000E-01	2	0.00	2.5000E-01	1	0.00
HS44	-1.5000E+01	3	0.00	-1.5000E+01	3	0.00	-1.5000E+01	3	0.00	-1.5000E+01	3	0.00	-1.5000E+01	2	0.00
HS44NEW	-1.5000E+01	5	0.00	-3.0000E+00	3	0.00	-1.5000E+01	5	0.00	-1.5000E+01	5	0.00	-1.5000E+01	4	0.00

1 = problem declared unbounded, 2 = problem declared infeasible, i = problem declared indefinite, f = failed, n = hit iteration limit

Table 8.3: Results for CUTEr QPs (continued)

	LUSOL			MA57			UMFPACK			MA57v			SQOPT		
Name	Objective	nItn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time
HS51	-8.8818E-16	3	0.00	0.0000E+00	1	0.00	-8.8818E-16	3	0.00	1.7764E-15	3	0.00	-8.8818E-16	2	0.00
HS52	5.3266E+00	3	0.00	5.3266E+00	1	0.00	5.3266E+00	3	0.00	5.3266E+00	3	0.00	5.3266E+00	2	0.00
HS53	4.0930E+00	3	0.00	4.0930E+00	1	0.00	4.0930E+00	3	0.00	4.0930E+00	3	0.00	4.0930E+00	2	0.00
HS76	-4.6818E+00	5	0.00	-4.6818E+00	5	0.00	-4.6818E+00	5	0.00	-4.6818E+00	5	0.00	-4.6818E+00	4	0.00
HS76I	-4.6818E+00	5	0.00	-4.6818E+00	5	0.00	-4.6818E+00	5	0.00	-4.6818E+00	5	0.00	-4.6818E+00	4	0.00
HUES-MOD	3.4830E+07	9304	21.52	3.4824E+07	559	1.74	3.4830E+07	9304	21.66	3.4830E+07	9304	24.53	3.4830E+07	8829	12.87
HUESTIS	2.3218E+14[f]	14	0.01	3.4824E+11	559	1.74	3.4830E+11	9304	21.58	3.4830E+11	9304	24.50	3.4824E+11	9700	14.90
KSIP	6.4220E+00[f]	481	0.28	5.7580E-01	253	0.75	5.7580E-01	2769	2.83	5.7580E-01	2769	5.37	5.7714E-01[i]	737	0.06
LINCONT	0.0000E+00[2]	138	0.03	0.0000E+00[2]	138	0.03	0.0000E+00[2]	138	0.03	0.0000E+00[2]	138	0.03	0.0000E+00[2]	138	0.03
LISWET1	3.6121E+01	3	0.59	3.6121E+01	1	0.23	3.6121E+01	3	0.61	3.6120E+01	3	0.44	3.6121E+01	26	0.02
LISWET10	4.9483E+01	41	3.61	4.9483E+01	22	0.73	4.9483E+01	41	3.84	4.9483E+01	46	2.71	4.9483E+01	96	0.07
LISWET11	4.9524E+01	50	2.42	4.9524E+01	45	1.72	4.9524E+01	50	2.59	4.9524E+01	50	1.92	4.9524E+01	92	0.07
LISWET12	1.7369E+03	26	2.97	1.7369E+03	27	1.60	1.7369E+03	26	3.18	1.7369E+03	26	2.22	1.7369E+03	38	0.03
LISWET2	2.5000E+01	21	0.61	2.5000E+01	19	0.72	2.5000E+01	21	0.65	2.5000E+01	21	0.51	2.5000E+01	138	0.10
LISWET3	2.5000E+01	434	2.86	2.5000E+01	436	6.28	2.5000E+01	434	4.83	2.5000E+01	430	5.08	2.5000E+01	778	0.56
LISWET4	2.5000E+01	426	3.11	2.5000E+01	424	5.77	2.5000E+01	426	4.63	2.5000E+01	414	4.39	2.5000E+01	822	0.73
LISWET5	2.5000E+01	409	2.77	2.5000E+01	407	5.33	2.5000E+01	409	3.95	2.5000E+01	413	4.58	2.5000E+01	794	0.71
LISWET6	2.5000E+01	337	3.47	2.5000E+01	335	5.34	2.5000E+01	337	4.38	2.4997E+01	408	7.06	2.5000E+01	645	0.46
LISWET7	4.9884E+02	3	0.59	4.9884E+02	1	0.23	4.9884E+02	3	0.63	4.9884E+02	3	0.44	4.9884E+02	26	0.02
LISWET8	7.1447E+02	22	1.82	7.1447E+02	19	0.72	7.1447E+02	22	1.91	7.1447E+02	22	1.36	7.1447E+02	53	0.04
LISWET9	1.9632E+03	21	2.99	1.9632E+03	18	1.14	1.9632E+03	21	3.18	1.9632E+03	23	2.42	1.9632E+03	38	0.03
LOTSCHD	2.3984E+03	9	0.00	2.3984E+03	9	0.00	2.3984E+03	9	0.00	2.3984E+03	9	0.00	2.3984E+03	8	0.00
MOSARQP1	NaN	2268	2.23	-3.8214E+03	1497	2.76	-3.8214E+03	3667	4.46	-3.8214E+03	3667	4.35	-3.8214E+03	3917	6.06
MOSARQP2	-5.0526E+03	2552	2.78	-5.0526E+03	850	1.50	-5.0526E+03	2552	2.88	-5.0526E+03	2552	3.09	-5.0526E+03	2591	6.32
NASH	0.0000E+00[2]	5	0.00	0.0000E+00[2]	5	0.00	0.0000E+00[2]	5	0.00	0.0000E+00[2]	5	0.00	0.0000E+00[2]	5	0.00
NCVXQP1	-7.1572E+07	755	0.62	-7.1587E+07	755	2.37	-7.1587E+07	763	0.95	-7.1587E+07	755	2.34	-2.2485E+06[i]	137	0.01
NCVXQP2	-5.7761E+07	1054	0.96	-5.7746E+07	1182	9.88	-5.7749E+07	1052	1.78	-5.7746E+07	1182	9.59	2.6397E+04[i]	139	0.01

1 = problem declared unbounded, 2 = problem declared infeasible, i = problem declared indefinite, f = failed, n = hit iteration limit

Table 8.3: Results for CUTEr QPs (continued)

	LUSOL			MA57			UMFPACK			MA57v			SQOPT		
Name	Objective	nItn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time
NCVXQP3	-2.9886E+07	1256	2.13	-2.9981E+07	1392	26.51	-2.9886E+07	1281	3.41	-2.9183E+07	1237	10.04	1.3057E+05 i	305	0.02
NCVXQP4	-9.3999E+07	777	0.07	-9.3999E+07	780	0.16	-9.3999E+07	777	0.08	-9.3999E+07	780	0.13	-1.3326E+06 i	22	0.00
NCVXQP5	-6.6260E+07	804	0.09	-6.6260E+07	804	0.19	-6.6260E+07	804	0.11	-6.6260E+07	804	0.16	-6.1834E+05 i	22	0.00
NCVXQP6	-3.3733E+07	909	0.11	-3.5137E+07	1184	5.60	-3.3733E+07	910	0.13	-3.3733E+07	904	0.29	4.1769E+05 i	74	0.00
NCVXQP7	-4.3523E+07	827	3.58	-4.3523E+07	828	17.55	-4.3523E+07	880	8.98	-4.3523E+07	828	17.25	-2.4070E+06 i	359	0.02
NCVXQP8	-3.0121E+07	919	4.21	-3.0117E+07	980	26.16	-3.0117E+07	930	9.74	-3.0117E+07	980	25.86	-1.1612E+06 i	359	0.02
NCVXQP9	-2.1146E+07	1063	7.04	-2.1146E+07	1134	32.65	-2.1146E+07	1044	14.35	-2.1146E+07	1134	32.44	2.6170E+05 i	447	0.03
PORTSNQP	3.3318E+03	10882	0.32	3.3318E+03	10882	0.32	3.3318E+03	10882	0.32	3.3318E+03	10882	0.32	3.3318E+03	10883	0.32
PORTSQP	3.3314E+03	10100	0.13	3.3314E+03	10099	0.13	3.3314E+03	10100	0.14	3.3314E+03	10100	0.13	3.3314E+03	10102	0.15
POWELL20	5.2090E+10	5002	25.50	5.2090E+10	5000	31.76	5.2090E+10	5002	28.81	5.2090E+10	5002	42.42	5.2090E+10	5005	2.68
PRIMAL1	-3.5013E-02	217	0.04	-3.5013E-02	70	0.03	-3.5013E-02	217	0.04	-3.5013E-02	216	0.05	-3.5013E-02	248	0.02
PRIMAL2	-3.3734E-02	408	0.09	-3.3734E-02	97	0.05	-3.3734E-02	408	0.12	-3.3734E-02	408	0.11	-3.3734E-02	423	0.06
PRIMAL3	-1.3576E-01	711	0.27	-1.3576E-01	102	0.10	-1.3576E-01	711	0.29	-1.3576E-01	711	0.35	-1.3576E-01	1258	0.31
PRIMAL4	-7.4609E-01	1223	0.52	-7.4609E-01	63	0.06	-7.4609E-01	1223	0.63	-7.4609E-01	1223	0.69	-7.4609E-01	1597	0.81
PRIMALC1	-6.1553E+03	19	0.00	-6.1553E+03	5	0.00	-6.1553E+03	19	0.00	-6.1553E+03	19	0.00	-6.1553E+03	20	0.00
PRIMALC2	-3.5513E+03	4	0.00	-3.5513E+03	4	0.00	-3.5513E+03	4	0.00	-3.5513E+03	4	0.00	-3.5513E+03	3	0.00
PRIMALC5	-4.2723E+02	11	0.00	-4.2723E+02	6	0.00	-4.2723E+02	11	0.00	-4.2723E+02	11	0.00	-4.2723E+02	13	0.00
PRIMALC8	-1.8309E+04	29	0.00	-1.8309E+04	9	0.00	-1.8309E+04	29	0.00	-1.8309E+04	29	0.00	-1.8309E+04	25	0.00
QPBAND	-9.9992E+03	29959	55.21	-9.9992E+03	29959	181.80	-9.9992E+03	29959	59.40	-9.9992E+03	29959	181.34	-9.9992E+03	26951	8.75
QPCBLEND	NaN	35	0.00	-7.8425E-03	96	0.01	-7.8425E-03	96	0.01	-7.8425E-03	96	0.01	-7.8425E-03	155	0.00
QPCBOEI1	1.1504E+07	1108	0.19	1.1504E+07	1106	0.46	1.1504E+07	1106	0.22	1.1504E+07	1106	0.45	1.1504E+07	1333	0.12
QPCBOEI2	8.1720E+06	287	0.02	8.1720E+06	288	0.04	8.1720E+06	289	0.03	8.1720E+06	288	0.04	8.1720E+06	245	0.01
QPCSTAIR	NaN	275	0.07	6.2044E+06	443	0.15	6.2044E+06	442	0.19	6.2044E+06	443	0.15	6.2044E+06	576	0.06
QPNBAND	-4.9997E+04	15000	26.63	-4.9997E+04	15000	39.05	-4.9997E+04	15000	29.10	-4.9997E+04	15000	39.07	-1.1249E+04 i	2	0.01
QPNBLEND	NaN	35	0.00	-8.7056E-03	83	0.01	-8.7056E-03	83	0.01	-8.7056E-03	83	0.01	-1.5705E-03 i	70	0.00
QPNBOEI1	6.7574E+06	1033	0.14	6.7574E+06	1035	0.35	6.7574E+06	1024	0.16	6.7574E+06	1035	0.35	8.7991E+06 i	768	0.05
QPNBOEI2	1.3683E+06	263	0.02	1.3683E+06	261	0.03	1.3683E+06	261	0.02	1.3683E+06	261	0.03	1.7260E+06 i	174	0.00

1 = problem declared unbounded, 2 = problem declared infeasible, i = problem declared indefinite, f = failed, n = hit iteration limit

Table 8.3: Results for CUTEr QPs (continued)

	LUSOL			MA57			UMFPACK			MA57v			SQOPT		
Name	Objective	nItn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time
QPNSTAIR	NaN	317	0.13	5.1460E+06	472	0.20	5.1460E+06	471	0.28	5.1460E+06	472	0.20	5.1460E+06[i]	564	0.06
S268	-3.6380E-12	7	0.00	-3.6380E-12	3	0.00	3.6380E-12	6	0.00	-1.0914E-11	6	0.00	0.0000E+00	8	0.00
SOSQP1	-2.4500E-11	2	0.53	-3.7823E-11	2	1.02	-2.4500E-11	2	0.52	-2.4500E-11	2	0.44	5.6357E-14	1	0.01
SOSQP2	-4.9987E+03	18540	55.98	-4.9987E+03	19252	749.04	-4.9987E+03	18540	60.82	-4.9987E+03	18540	240.94	-4.9987E+03	18462	43.30
STATIC3	-3.0892E+02[1]	3	0.00	-2.5298E+03[1]	12	0.01	-3.0892E+02[1]	3	0.00	-3.0892E+02[1]	3	0.00	-6.3723E+02[i]	6	0.00
STCQP1	3.6710E+05	7266	16.09	3.6710E+05	1550	9.45	3.6710E+05	7266	18.88	3.6710E+05	7266	56.22	3.6710E+05	7391	34.87
STCQP2	3.7189E+04	7589	16.74	3.7189E+04	3279	2.59	3.7189E+04	7589	15.25	3.7189E+04	7590	38.26	3.7189E+04	7684	21.10
STEENBRA	1.6958E+04	86	0.01	1.6958E+04	87	0.01	1.6958E+04	87	0.01	1.6958E+04	87	0.01	1.6958E+04	101	0.00
STNQP1	-3.1170E+05	7249	15.29	-3.1170E+05	2101	62.94	-3.1170E+05	7249	17.88	-3.1170E+05	7249	51.52	-2.3135E+05[i]	828	0.23
STNQP2	-5.7497E+05	7249	8.40	-5.7497E+05	4963	455.35	-5.7497E+05	7249	9.31	-5.7497E+05	7249	17.90	-1.4154E+05[i]	3152	0.59
TAME	0.0000E+00	2	0.00	0.0000E+00	2	0.00	0.0000E+00	2	0.00	0.0000E+00	2	0.00	3.0815E-33	1	0.00
UBH1	NaN[2]	784	1.08	1.1160E+00	7765	128.97	1.1160E+00	9893	145.29	1.1160E+00	10316	161.19	3.3482E+01[i]	1852	2.10
WALL10	0.0000E+00	1	0.00	0.0000E+00	1	0.03	0.0000E+00	1	0.00	0.0000E+00	1	0.00	0.0000E+00	0	0.00
WALL100	0.0000E+00	1	0.03	0.0000E+00	1	275.28	0.0000E+00	1	0.03	0.0000E+00	1	0.03	0.0000E+00	0	0.01
WALL20	0.0000E+00	1	0.00	0.0000E+00	1	0.41	0.0000E+00	1	0.00	0.0000E+00	1	0.00	0.0000E+00	0	0.00
WALL50	0.0000E+00	1	0.01	0.0000E+00	1	16.19	0.0000E+00	1	0.01	0.0000E+00	1	0.01	0.0000E+00	0	0.00
YAO	1.9770E+02	3	0.02	1.9770E+02	3	0.02	1.9770E+02	3	0.02	1.9770E+02	3	0.02	1.9770E+02	12	0.00
ZECEVIC2	-4.1250E+00	5	0.00	-4.1250E+00	5	0.00	-4.1250E+00	5	0.00	-4.1250E+00	5	0.00	-4.1250E+00	4	0.00

1 = problem declared unbounded, 2 = problem declared infeasible, i = problem declared indefinite, f = failed, n = hit iteration limit

Table 8.4: Results for Maros and Mészáros QPs

	LUSOL			MA57			UMFPACK			MA57v			SQOPT		
Name	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time
ADAT1	-2.8527E+07	11	0.03	-2.8527E+07	20	3.84	-2.8527E+07	11	0.03	-2.8527E+07	6	0.04	-2.8527E+07	28	0.02
ADAT2	-3.2627E+01	32	0.11	-3.2627E+01	32	7.00	-3.2627E+01	30	0.07	-3.2627E+01	28	0.09	-3.2627E+01	42	0.02
ADAT3	-3.5779E+01	19	0.38	-3.5779E+01	19	16.16	-3.5779E+01	21	0.17	-3.5779E+01	30	0.27	-3.5779E+01	26	0.02
AUG2D	1.6874E+06	10193	74.33	1.6874E+06	396	3.27	1.6874E+06	10193	92.92	1.6874E+06	10193	86.79	1.6874E+06	10615	560.15
AUG2DC	1.8184E+06	10201	77.11	1.8184E+06	1	0.87	1.8184E+06	10201	97.58	1.8184E+06	10201	86.84	1.8184E+06	10622	556.76
AUG2DCQP	6.4981E+06	14479	94.39	6.4981E+06	14334	133.70	6.4981E+06	14361	117.86	6.4981E+06	14334	133.87	6.4981E+06	14472	569.74
AUG2DQP	6.2370E+06	14599	93.44	6.2370E+06	14591	133.51	6.2370E+06	14266	114.70	6.2370E+06	14591	133.57	6.2370E+06	14185	560.27
AUG3D	5.5407E+02	2159	6.52	5.5407E+02	484	0.73	5.5407E+02	2159	6.40	5.5407E+02	2159	6.05	5.5407E+02	2356	5.48
AUG3DC	7.7126E+02	2874	10.40	7.7126E+02	1	0.04	7.7126E+02	2874	10.52	7.7126E+02	2874	8.70	7.7126E+02	2873	8.18
AUG3DCQP	9.9336E+02	2803	6.56	9.9336E+02	2794	6.91	9.9336E+02	2796	6.93	9.9336E+02	2794	6.92	9.9336E+02	2876	6.25
AUG3DQP	6.7524E+02	1954	5.51	6.7524E+02	1940	5.54	6.7524E+02	1947	5.70	6.7524E+02	1940	5.54	6.7524E+02	1948	1.78
CONT-050	-4.5639E+00	1241	1.18	-4.5639E+00	1241	1.51	-4.5639E+00	1241	1.30	-4.5639E+00	1241	1.51	-4.5639E+00	1242	0.95
CONT-100	-4.6444E+00	1882	18.29	-4.6444E+00	1882	24.82	-4.6444E+00	1882	17.91	-4.6444E+00	1882	24.74	-4.6444E+00	2033	12.60
CONT-101	1.9553E-01	1094	10.40	1.9553E-01	1094	11.84	1.9553E-01	1094	10.01	1.9553E-01	1094	11.84	1.9553E-01	1101	8.98
CONT-200	-4.6849E+00	2638	212.80	-4.6849E+00	2638	321.94	-4.6849E+00	2638	205.19	-4.6849E+00	2638	322.32	-4.6849E+00	3156	180.46
CONT-201	1.9248E-01	2223	180.05	1.9248E-01	2223	202.90	1.9248E-01	2223	172.42	1.9248E-01	2223	202.58	1.9248E-01	2188	164.47
CONT-300	1.9151E-01	3448	1432.41	1.9151E-01	3451	1116.22	1.9151E-01	3448	1057.25	1.9151E-01	3451	1115.30	1.9151E-01	3430	972.09
CVXQP1_L	4.4406E+08[2]	4360	2.57	1.9023E+09[2]	4363	51.09	2.0071E+46[2]	4793	506.51	1.9023E+09[2]	4363	51.10	1.0870E+08	10837	11.21
CVXQP1_M	1.0875E+06	517	0.82	1.0875E+06	512	3.11	1.0875E+06	510	1.20	1.0875E+06	512	3.11	1.0875E+06	667	0.06
CVXQP1_S	1.1591E+04	38	0.00	1.1591E+04	39	0.00	1.1591E+04	39	0.00	1.1591E+04	39	0.00	1.1591E+04	39	0.00
CVXQP2_L	NaN[2]	1246	17.55	8.1842E+07	3242	608.66	8.1842E+07	3215	88.97	8.1842E+07	3242	608.70	8.1842E+07	3634	10.83
CVXQP2_M	8.2016E+05	306	0.07	8.2016E+05	309	0.13	8.2016E+05	306	0.08	8.2016E+05	309	0.13	8.2016E+05	333	0.03
CVXQP2_S	8.1209E+03	30	0.00	8.1209E+03	31	0.00	8.1209E+03	30	0.00	8.1209E+03	31	0.00	8.1209E+03	31	0.00
CVXQP3_L	NaN[2]	8013	66.17	1.3218E+14[2]	8012	267.31	3.2944E+08[2]	8903	6800.25	1.3218E+14[2]	8012	267.26	1.1571E+08	10654	9.07
CVXQP3_M	1.3628E+06	601	3.71	1.3628E+06	602	13.32	1.3628E+06	579	6.54	1.3628E+06	602	13.32	1.3628E+06	624	0.05
CVXQP3_S	1.1943E+04	28	0.00	1.1943E+04	29	0.00	1.1943E+04	30	0.00	1.1943E+04	29	0.00	1.1943E+04	32	0.00
DPKLO1	3.7010E-01	57	0.00	3.7010E-01	57	0.00	3.7010E-01	57	0.00	3.7010E-01	57	0.00	3.7010E-01	56	0.00

1 = problem declared unbounded, 2 = problem declared infeasible, i = problem declared indefinite, f = failed, n = hit iteration limit

Table 8.4: Results for Maros and Mészáros QPs (continued)

	LUSOL			MA57			UMFPACK			MA57v			SQOPT		
Name	Objective	nItn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time
DTOC3	2.3526E+02	4806	83.55	2.3526E+02	3	0.48	2.3526E+02	4806	152.41	2.3526E+02	4806	189.80	2.3526E+02	4805	11.79
DUAL1	3.5013E-02	89	0.00	3.5013E-02	89	0.00	3.5013E-02	89	0.01	3.5013E-02	89	0.00	3.5013E-02	82	0.00
DUAL2	3.3734E-02	100	0.01	3.3734E-02	100	0.01	3.3734E-02	100	0.01	3.3734E-02	100	0.01	3.3734E-02	99	0.00
DUAL3	1.3576E-01	107	0.01	1.3576E-01	107	0.01	1.3576E-01	107	0.01	1.3576E-01	107	0.01	1.3576E-01	118	0.01
DUAL4	7.4609E-01	62	0.00	7.4609E-01	62	0.00	7.4609E-01	62	0.00	7.4609E-01	62	0.00	7.4609E-01	67	0.00
DUALC1	6.1553E+03	10	0.00	6.1553E+03	10	0.02	6.1553E+03	10	0.00	6.1553E+03	10	0.02	6.1553E+03	9	0.00
DUALC2	3.5513E+03	5	0.00	3.5513E+03	5	0.02	3.5513E+03	5	0.00	3.5513E+03	5	0.02	3.5513E+03	4	0.00
DUALC5	4.2723E+02	8	0.00	4.2723E+02	8	0.03	4.2723E+02	8	0.00	4.2723E+02	8	0.03	4.2723E+02	7	0.00
DUALC8	1.8309E+04	7	0.00	1.8309E+04	7	0.12	1.8309E+04	7	0.00	1.8309E+04	7	0.12	1.8309E+04	8	0.00
EXDATA	-1.4184E+02	2245	33.28	-1.4184E+02	2304	60.25	-1.4184E+02	2245	93.09	-1.4184E+02	2245	61.36	-1.4184E+02	2320	14.65
GENHS28	9.2717E-01	3	0.00	9.2717E-01	1	0.00	9.2717E-01	3	0.00	9.2717E-01	3	0.00	9.2717E-01	2	0.00
GOULDQP2	1.8427E-04	343	0.07	1.8427E-04	343	0.10	1.8427E-04	343	0.08	1.8427E-04	343	0.10	1.8427E-04	342	0.03
GOULDQP3	2.0628E+00	428	0.05	2.0628E+00	428	0.11	2.0628E+00	428	0.06	2.0628E+00	428	0.11	2.0628E+00	445	0.02
HS118	6.6482E+02	20	0.00	6.6482E+02	20	0.00	6.6482E+02	20	0.00	6.6482E+02	20	0.00	6.6482E+02	19	0.00
HS21	-9.9960E+01	1	0.00	-9.9960E+01	1	0.00	-9.9960E+01	1	0.00	-9.9960E+01	1	0.00	-9.9960E+01	0	0.00
HS268	-9.0949E-12	7	0.00	-5.4570E-12	2	0.00	0.0000E+00	7	0.00	-1.6371E-11	7	0.00	5.4570E-12	9	0.00
HS35	1.1111E-01	5	0.00	1.1111E-01	5	0.00	1.1111E-01	5	0.00	1.1111E-01	5	0.00	1.1111E-01	4	0.00
HS35MOD	2.5000E-01	3	0.00	2.5000E-01	3	0.00	2.5000E-01	3	0.00	2.5000E-01	3	0.00	2.5000E-01	2	0.00
HS51	0.0000E+00	3	0.00	0.0000E+00	1	0.00	0.0000E+00	3	0.00	0.0000E+00	3	0.00	0.0000E+00	2	0.00
HS52	5.3266E+00	3	0.00	5.3266E+00	1	0.00	5.3266E+00	3	0.00	5.3266E+00	3	0.00	5.3266E+00	2	0.00
HS53	4.0930E+00	3	0.00	4.0930E+00	1	0.00	4.0930E+00	3	0.00	4.0930E+00	3	0.00	4.0930E+00	2	0.00
HS76	-4.6818E+00	4	0.00	-4.6818E+00	4	0.00	-4.6818E+00	4	0.00	-4.6818E+00	4	0.00	-4.6818E+00	3	0.00
HUES-MOD	3.4830E+07	8338	20.68	3.4830E+07	8338	23.70	3.4830E+07	8338	20.69	3.4830E+07	8338	23.67	3.4830E+07	8333	12.13
HUESTIS	2.3183E+14^f	12	0.01	3.4830E+11	8338	23.68	3.4830E+11	8338	20.71	3.4830E+11	8338	23.69	3.4824E+11	9184	13.96
KSIP	5.8038E-01^f	425	0.46	5.7580E-01	635	1.99	5.7580E-01	579	0.37	5.7580E-01	585	0.57	5.7580E-01	484	0.05
LASER	2.4096E+06	532	0.07	2.4096E+06	530	0.25	2.4096E+06	532	0.23	2.4096E+06	532	0.27	2.4096E+06	531	0.04
LISWET1	3.6122E+01	3	0.59	3.6122E+01	1	0.22	3.6122E+01	3	0.61	3.6122E+01	3	0.44	3.6122E+01	25	0.02

1 = problem declared unbounded, 2 = problem declared infeasible, i = problem declared indefinite, f = failed, n = hit iteration limit

Table 8.4: Results for Maros and Mészáros QPs (continued)

	LUSOL			MA57			UMFPACK			MA57v			SQOPT		
Name	Objective	nItn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time
LISWET10	4.9486E+01	41	3.61	4.9486E+01	22	0.72	4.9486E+01	41	3.83	4.9486E+01	46	2.72	4.9486E+01	98	0.07
LISWET11	4.9524E+01	50	2.41	4.9524E+01	45	1.71	4.9524E+01	50	2.59	4.9524E+01	50	1.93	4.9524E+01	92	0.07
LISWET12	1.7369E+03	26	2.98	1.7369E+03	27	1.59	1.7369E+03	26	3.17	1.7369E+03	26	2.22	1.7369E+03	38	0.03
LISWET2	2.4998E+01	21	0.61	2.4998E+01	19	0.72	2.4998E+01	21	0.64	2.4998E+01	21	0.51	2.4998E+01	138	0.10
LISWET3	2.5001E+01	438	3.01	2.5001E+01	436	6.29	2.5001E+01	438	4.86	2.5001E+01	430	5.09	2.5001E+01	783	0.55
LISWET4	2.5000E+01	426	3.72	2.5000E+01	424	5.78	2.5000E+01	426	4.63	2.5000E+01	414	4.39	2.5000E+01	828	0.74
LISWET5	2.5034E+01	409	2.76	2.5034E+01	405	5.62	2.5034E+01	409	4.24	2.5034E+01	413	4.72	2.5034E+01	786	0.70
LISWET6	2.4996E+01	337	3.68	2.4996E+01	335	5.35	2.4996E+01	337	4.37	2.4993E+01	408	7.09	2.4996E+01	645	0.57
LISWET7	4.9884E+02	3	0.59	4.9884E+02	1	0.22	4.9884E+02	3	0.63	4.9884E+02	3	0.44	4.9884E+02	24	0.02
LISWET8	7.1447E+02	22	1.82	7.1447E+02	19	0.71	7.1447E+02	22	1.91	7.1447E+02	22	1.36	7.1447E+02	54	0.04
LISWET9	1.9633E+03	21	2.98	1.9633E+03	18	1.13	1.9633E+03	21	3.17	1.9633E+03	23	2.42	1.9633E+03	37	0.03
LOTSCHD	2.3984E+03	9	0.00	2.3984E+03	9	0.00	2.3984E+03	9	0.00	2.3984E+03	9	0.00	2.3984E+03	8	0.00
MOSARQP1	-9.3423E+02^{2}	1630	1.70	-9.5288E+02	7305	20.42	-9.5288E+02	7220	25.11	-9.5288E+02	7305	20.43	-9.5288E+02	3373	3.87
MOSARQP2	NaN^{2}	1022	0.37	-1.5975E+03	1515	0.87	-1.5975E+03	1515	0.78	-1.5975E+03	1515	0.87	-1.5975E+03	2306	0.72
POWELL20	5.2090E+10	5002	25.50	5.2090E+10	5000	31.53	5.2090E+10	5002	28.80	5.2090E+10	5002	42.43	5.2090E+10	5005	2.67
PRIMAL1	-3.5013E-02	217	0.04	-3.5013E-02	70	0.03	-3.5013E-02	217	0.04	-3.5013E-02	216	0.05	-3.5013E-02	245	0.02
PRIMAL2	-3.3734E-02	408	0.09	-3.3734E-02	97	0.05	-3.3734E-02	408	0.12	-3.3734E-02	408	0.11	-3.3734E-02	425	0.06
PRIMAL3	-1.3576E-01	711	0.27	-1.3576E-01	102	0.10	-1.3576E-01	711	0.29	-1.3576E-01	711	0.35	-1.3576E-01	1256	0.31
PRIMAL4	-7.4609E-01	1223	0.52	-7.4609E-01	63	0.06	-7.4609E-01	1223	0.63	-7.4609E-01	1223	0.69	-7.4609E-01	1592	0.81
PRIMALC1	-6.1553E+03	19	0.00	-6.1553E+03	5	0.00	-6.1553E+03	19	0.00	-6.1553E+03	19	0.00	-6.1553E+03	20	0.00
PRIMALC2	-3.5513E+03	4	0.00	-3.5513E+03	4	0.00	-3.5513E+03	4	0.00	-3.5513E+03	4	0.00	-3.5513E+03	3	0.00
PRIMALC5	-4.2723E+02	11	0.00	-4.2723E+02	6	0.00	-4.2723E+02	11	0.00	-4.2723E+02	11	0.00	-4.2723E+02	13	0.00
PRIMALC8	-1.8309E+04	29	0.00	-1.8309E+04	9	0.00	-1.8309E+04	29	0.00	-1.8309E+04	29	0.00	-1.8309E+04	25	0.00
Q25FV47	1.5929E+07^{1}	5127	10.03	1.4710E+07^{f}	6052	12.43	1.3744E+07	8556	13.05	1.4710E+07^{f}	6052	12.43	1.3744E+07^{i}	11555	9.01
QADLITTL	4.8032E+05	148	0.00	4.8032E+05	148	0.01	4.8032E+05	148	0.01	4.8032E+05	148	0.01	4.8066E+05^{i}	156	0.00
QAFIRO	-1.5908E+00	10	0.00	-1.5908E+00	10	0.00	-1.5908E+00	10	0.00	-1.5908E+00	10	0.00	-1.5908E+00	9	0.00
QBANDM	1.6352E+04	615	0.05	1.6352E+04	615	0.09	1.6352E+04	615	0.06	1.6352E+04	615	0.09	1.6401E+04^{i}	382	0.02

1 = problem declared unbounded, 2 = problem declared infeasible, i = problem declared indefinite, f = failed, n = hit iteration limit

Table 8.4: Results for Maros and Mészáros QPs (continued)

	LUSOL			MA57			UMFPACK			MA57v			SQOPT		
Name	Objective	nItn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time
QBEACONF	1.6471E+05	45	0.00	1.6471E+05	45	0.01	1.6471E+05	45	0.00	1.6471E+05	45	0.01	1.6471E+05	44	0.00
QBORE3D	3.1002E+03	111	0.01	3.1002E+03	111	0.01	3.1002E+03	111	0.01	3.1002E+03	111	0.01	3.1002E+03	118	0.00
QBRANDY	2.8375E+04	404	0.02	2.8375E+04	399	0.04	2.8375E+04	399	0.03	2.8375E+04	399	0.04	2.8375E+04	412	0.02
QCAPRI	6.6793E+07	251	0.02	6.6793E+07	258	0.05	6.6793E+07	252	0.02	6.6793E+07	258	0.05	6.6793E+07	264	0.01
QE226	2.1265E+02	687	0.08	2.1265E+02	688	0.13	2.1265E+02	688	0.12	2.1265E+02	688	0.13	2.5375E+02^{i}	186	0.01
QETAMACR	8.6760E+04	786	0.15	8.6760E+04	798	0.32	8.6760E+04	772	0.18	8.6760E+04	798	0.33	8.6760E+04	734	0.05
QFFFFF80	8.7315E+05	723	0.15	8.7315E+05	734	0.30	8.7315E+05	732	0.17	8.7315E+05	734	0.30	9.1244E+05^{i}	640	0.05
QFORPLAN	7.4566E+09	185	0.01	7.4566E+09	188	0.03	7.4566E+09	185	0.02	7.4566E+09	188	0.03	7.4566E+09	161	0.01
QGFRDXPN	1.0079E+11	578	0.07	1.0079E+11	577	0.18	1.0079E+11	578	0.09	1.0079E+11	577	0.18	1.0079E+11^{i}	408	0.02
QGROW15	-1.0169E+08	568	0.31	-1.0169E+08	493	0.32	-1.0169E+08	538	0.29	-1.0169E+08	493	0.32	-9.7817E+07^{i}	526	0.08
QGROW22	-1.4963E+08	907	0.53	-1.4963E+08	948	0.66	-1.4963E+08	1011	0.78	-1.4963E+08	948	0.66	-1.0792E+04^{1}	350	0.06
QGROW7	-4.2799E+07	231	0.04	-4.2799E+07	272	0.06	-4.2799E+07	233	0.05	-4.2799E+07	272	0.06	-4.1431E+07^{i}	349	0.03
QISRAEL	2.5348E+07	199	0.01	2.5348E+07	212	0.03	2.5348E+07	202	0.02	2.5348E+07	212	0.03	2.5348E+07^{i}	113	0.00
QPCBLEND	NaN	35	0.00	-7.8425E-03	96	0.01	-7.8425E-03	96	0.01	-7.8425E-03	96	0.01	-7.8425E-03	155	0.00
QPCBOEI1	1.1504E+07	1100	0.18	1.1504E+07	1113	0.42	1.1504E+07	1094	0.21	1.1504E+07	1113	0.42	1.1504E+07	1453	0.15
QPCBOEI2	8.1720E+06	230	0.01	8.1720E+06	229	0.03	8.1720E+06	230	0.02	8.1720E+06	229	0.03	8.1720E+06	254	0.01
QPCSTAIR	6.2044E+06	446	0.17	6.2044E+06	444	0.17	6.2044E+06	440	0.22	6.2044E+06	444	0.17	6.2044E+06	564	0.06
QPILOTNO	4.7328E+06^{f}	7055	1.01	7.8628E+06^{f}	6845	0.76	NaN^{2}	7993	2.95	7.8628E+06^{f}	6845	0.76	4.7317E+06^{i}	56199	69.61
QPTEST	4.3719E+00	2	0.00	4.3719E+00	2	0.00	4.3719E+00	2	0.00	4.3719E+00	2	0.00	4.3719E+00	1	0.00
QRECIPE	-2.6662E+02	27	0.00	-2.6662E+02	27	0.00	-2.6662E+02	27	0.00	-2.6662E+02	27	0.00	-2.6662E+02	27	0.00
QSC205	-5.8140E-03	21	0.00	-5.8140E-03	21	0.00	-5.8140E-03	21	0.00	-5.8140E-03	21	0.00	-5.8140E-03	21	0.00
QSCAGR25	2.0174E+08	832	0.06	2.0174E+08	832	0.13	2.0174E+08	832	0.09	2.0174E+08	832	0.13	2.2025E+08^{i}	500	0.03
QSCAGR7	2.6866E+07	125	0.00	2.6866E+07	129	0.01	2.6866E+07	129	0.01	2.6866E+07	129	0.01	2.7079E+07^{i}	93	0.00
QSCFXM1	1.6883E+07	375	0.03	1.6883E+07	374	0.06	1.6883E+07	375	0.04	1.6883E+07	374	0.06	1.6883E+07	468	0.02
QSCFXM2	2.7777E+07	745	0.14	2.7777E+07	743	0.26	2.7777E+07	743	0.16	2.7777E+07	743	0.26	2.7789E+07^{i}	598	0.04
QSCFXM3	3.0817E+07	1265	0.39	3.0817E+07	1266	0.65	3.0817E+07	1266	0.43	3.0817E+07	1266	0.65	3.0817E+07	1321	0.16
QSCORPIO	1.8805E+03	213	0.02	1.8805E+03	219	0.04	1.8805E+03	212	0.02	1.8805E+03	219	0.04	1.8805E+03	184	0.01

1 = problem declared unbounded, 2 = problem declared infeasible, i = problem declared indefinite, f = failed, n = hit iteration limit

Table 8.4: Results for Maros and Mészáros QPs (continued)

	LUSOL			MA57			UMFPACK			MA57v			SQOPT		
Name	Objective	nItn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time
QSCRS8	9.0456E+02	724	0.14	9.0456E+02	728	0.27	9.0456E+02	720	0.16	9.0456E+02	728	0.26	9.0456E+02^{n}	11690	?.??
QSCSD1	8.6667E+00	240	0.02	8.6667E+00	199	0.02	8.6667E+00	219	0.02	8.6667E+00	199	0.02	8.6667E+00	222	0.01
QSCSD6	5.0808E+01	571	0.08	5.0808E+01	659	0.12	5.0808E+01	643	0.10	5.0808E+01	659	0.12	5.8950E+01^{i}	320	0.01
QSCSD8	9.4076E+02	1193	0.37	9.4076E+02	1202	0.53	9.4076E+02	1244	0.40	9.4076E+02	1202	0.53	1.4054E+03^{i}	441	0.02
QSCTAP1	1.4159E+03	289	0.02	1.4159E+03	289	0.03	1.4159E+03	290	0.02	1.4159E+03	289	0.03	1.4159E+03	298	0.01
QSCTAP2	1.7350E+03	1300	1.03	1.7350E+03	1322	1.60	1.7350E+03	1293	1.05	1.7350E+03	1322	1.60	1.7350E+03	1374	0.23
QSCTAP3	1.4388E+03	1616	0.50	1.4388E+03	1597	1.35	1.4388E+03	1653	1.02	1.4388E+03	1597	1.35	2.2922E+03^{i}	1059	0.19
QSEBA	8.1483E+07	296	0.02	8.1483E+07	296	0.03	8.1483E+07	296	0.02	8.1483E+07	296	0.03	8.1483E+07	298	0.02
QSHARE1B	7.2008E+05	407	0.02	7.2008E+05	407	0.03	7.2008E+05	407	0.02	7.2008E+05	407	0.03	7.5304E+05^{i}	231	0.00
QSHARE2B	1.1704E+04	113	0.00	1.1704E+04	113	0.00	1.1704E+04	113	0.00	1.1704E+04	113	0.00	1.1704E+04	109	0.00
QSHELL	1.5726E+12	517	0.34	1.5726E+12	504	0.37	1.5726E+12	538	0.38	1.5726E+12	504	0.37	1.5925E+12^{i}	339	0.05
QSHIP04L	2.4200E+06	258	0.15	2.4200E+06	260	0.16	2.4200E+06	258	0.16	2.4200E+06	260	0.16	2.4227E+06^{i}	252	0.01
QSHIP04S	2.4250E+06	171	0.07	2.4250E+06	171	0.07	2.4250E+06	171	0.07	2.4250E+06	171	0.07	2.4250E+06	189	0.01
QSHIP08L	2.3760E+06	420	0.36	2.3760E+06	419	0.59	2.3760E+06	420	0.38	2.3760E+06	419	0.60	2.3764E+06^{i}	426	0.12
QSHIP08S	2.3857E+06	245	0.11	2.3857E+06	244	0.23	2.3857E+06	248	0.12	2.3857E+06	244	0.23	2.4105E+06^{i}	170	0.02
QSHIP12L	3.0189E+06	838	2.85	3.0189E+06	841	3.63	3.0189E+06	833	2.76	3.0189E+06	841	3.64	3.0336E+06^{i}	606	0.28
QSHIP12S	3.0570E+06	435	0.80	3.0570E+06	433	1.05	3.0570E+06	435	0.83	3.0570E+06	433	1.05	3.0595E+06^{i}	363	0.07
QSIERRA	2.3753E+07	590	0.28	2.3753E+07	588	0.42	2.3753E+07	590	0.31	2.3753E+07	588	0.42	2.4062E+07^{i}	552	0.04
QSTAIR	7.9855E+06	355	0.05	7.9855E+06	349	0.08	7.9855E+06	355	0.07	7.9855E+06	349	0.08	7.9855E+06	472	0.04
QSTANDAT	6.4118E+03	188	0.01	6.4118E+03	190	0.03	6.4118E+03	186	0.02	6.4118E+03	190	0.03	6.4628E+03^{i}	162	0.01
S268	-9.0949E-12	7	0.00	-5.4570E-12	2	0.00	0.0000E+00	7	0.00	-1.6371E-11	7	0.00	5.4570E-12	9	0.00
STCQP1	1.5514E+05	1042	0.37	1.5514E+05	779	1.49	1.5514E+05	1042	0.42	1.5514E+05	1042	1.22	1.5514E+05	1046	0.28
STCQP2	2.2327E+04	2384	1.23	2.2327E+04	1645	0.54	2.2327E+04	2384	1.30	2.2327E+04	2388	2.67	2.2327E+04	2401	0.51
TAME	0.0000E+00	2	0.00	0.0000E+00	2	0.00	0.0000E+00	2	0.00	0.0000E+00	2	0.00	3.0815E-33	1	0.00
UBH1	NaN^{2}	784	1.10	1.1160E+00	7765	129.44	1.1160E+00	9893	146.46	1.1160E+00	10316	161.48	3.3482E+01^{i}	1852	2.10
VALUES	-1.3966E+00	79	0.00	-1.3966E+00	79	0.00	-1.3966E+00	79	0.01	-1.3966E+00	79	0.00	-1.3966E+00	79	0.00
YAO	1.9770E+02	3	0.02	1.9770E+02	3	0.02	1.9770E+02	3	0.02	1.9770E+02	3	0.02	1.9770E+02	12	0.00

1 = problem declared unbounded, 2 = problem declared infeasible, i = problem declared indefinite, f = failed, n = hit iteration limit

Table 8.4: Results for Maros and Mészáros QPs (continued)

	LUSOL			MA57			UMFPACK			MA57v			SQOPT		
Name	Objective	nItn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time	Objective	Itn	Time
ZECEVIC2	-4.1250E+00	5	0.00	-4.1250E+00	5	0.00	-4.1250E+00	5	0.00	-4.1250E+00	5	0.00	-4.1250E+00	4	0.00

1 = problem declared unbounded, 2 = problem declared infeasible, i = problem declared indefinite, f = failed, n = hit iteration limit

Table 8.5: Number of superbasics and factorizations for CUTEr problems

	nS					nFac			
Name	lusol	ma57	umfpack	ma57v	sqopt	lusol	ma57	umfpack	ma57v
ALLINQP	1964	1964	1964	1964	1964	4	2	4	4
AUG2D	10192	10196	10192	10192	10192	11	1	11	11
AUG2DC	10200	10200	10200	10200	10200	11	1	11	11
AUG2DCQP	9994	9994	9994	9994	9994	18	18	18	18
AUG2DQP	9801	9801	9801	9801	9801	18	18	17	18
AUG3D	16909	16908	16909	16909	16909	17	3	17	17
AUG3DC	19543	19543	19543	19543	19543	20	1	20	20
AUG3DCQP	17665	17665	17665	17665	17665	25	25	25	25
AUG3DQP	13712	13712	13712	13712	13713	21	21	21	21
AVGASA	3	3	3	3	3	1	1	1	1
AVGASB	3	3	3	3	3	1	1	1	1
BIGGSC4	1	1	1	1	1	2	2	2	2
BLOCKQP1	9	9	9	9	2	1	2	1	1
BLOCKQP2	9	9	9	9	2002	1110	8	1110	1110
BLOCKQP3	9	9	9	9	2	1	2	1	1
BLOCKQP4	9	9	9	9	2002	1024	12	1024	1024
BLOCKQP5	9	9	9	9	0	1	2	1	1
BLOWEYA	0	2000	0	0	0	1	3	1	1
BLOWEYB	0	2000	0	0	0	1	3	1	1
BLOWEYC	0	2000	0	0	0	1	3	1	1
CONT5-QP	2	0	2	1	97	3	3	4	6
CVXQP1	0	675	0	0	1275	1	6	378	2
CVXQP2	2210	2210	2210	2210	2210	437	93	423	440
CVXQP3	0	1758	0	0	436	1	2	5	2
DEGENQP	0	0	0	0	0	1	1	1	1
DTOC3	4803	4999	4803	4803	4803	5	1	5	5
DUAL1	62	62	62	62	62	14	14	14	14
DUAL2	91	91	91	91	91	15	15	15	15
DUAL3	96	96	96	96	96	15	15	15	15
DUAL4	61	61	61	61	61	13	13	13	13
DUALC1	2	2	2	2	2	1	1	1	1
DUALC2	2	2	2	2	2	1	1	1	1
DUALC5	4	4	4	4	4	1	1	1	1
DUALC8	2	2	2	2	2	1	1	1	1
FERRISDC	0	206	0	0	0	1	1	1	1
GENHS28	2	2	2	2	2	1	1	1	1
GMNCASE1	51	95	51	51	51	1	2	1	1
GMNCASE2	46	94	46	46	46	3	1	3	3
GMNCASE3	48	93	48	48	48	3	7	3	3
GMNCASE4	0	0	0	0	0	1	1	1	1
GOULDQP2	0	0	0	0	0	1	1	1	1
GOULDQP3	4988	4988	4988	4988	4988	6	6	6	6

Table 8.5: Number of superbasics and factorizations for CUTEr problems (continued)

	nS					nFac			
Name	lusol	ma57	umfpack	ma57v	sqopt	lusol	ma57	umfpack	ma57v
GRIDNETA	2183	2218	2183	2183	2183	3	1	3	3
GRIDNETB	6560	6561	6560	6560	6561	7	1	7	7
GRIDNETC	4533	4533	4533	4533	4533	5	3	5	5
HATFLDH	0	0	0	0	0	1	2	1	1
HS118	0	0	0	0	0	2	1	2	2
HS21	1	1	1	1	1	3	2	3	3
HS268	4	5	5	5	5	2	1	2	2
HS35	2	2	2	2	2	6	2	6	6
HS35I	2	2	2	2	2	6	2	6	6
HS35MOD	1	2	1	1	1	3	1	3	3
HS44	0	0	0	0	0	1	1	1	1
HS44NEW	0	1	0	0	0	1	1	1	1
HS51	2	2	2	2	2	1	1	1	1
HS52	2	2	2	2	2	1	1	1	1
HS53	2	2	2	2	2	1	1	1	1
HS76	2	2	2	2	2	2	1	2	2
HS76I	2	2	2	2	2	2	1	2	2
HUES-MOD	8321	9444	8321	8321	8323	27	1	27	27
HUESTIS	3	9444	8321	8321	9138	4	1	27	27
KSIP	1	18	18	18	16	121	112	1279	1279
LINCONT	0	0	0	0	0	4	4	4	4
LISWET1	2	2	2	2	2	2	1	2	2
LISWET10	14	17	14	17	15	12	3	12	12
LISWET11	31	36	31	31	31	8	7	8	8
LISWET12	5	6	5	5	5	10	7	10	10
LISWET2	4	4	4	4	16	2	3	2	2
LISWET3	261	261	261	261	282	7	11	7	8
LISWET4	269	269	269	269	284	8	13	8	7
LISWET5	254	254	254	254	265	7	11	7	8
LISWET6	222	222	222	231	239	10	15	10	21
LISWET7	2	2	2	2	2	2	1	2	2
LISWET8	13	14	13	13	15	6	3	6	6
LISWET9	4	5	4	4	4	10	5	10	11
LOTSCHD	0	0	0	0	0	1	1	1	1
MOSARQP1	909	1021	1021	1021	1021	7	2	147	144
MOSARQP2	1640	1640	1640	1640	1640	3	1	3	3
NASH	0	0	0	0	0	2	2	2	2
NCVXQP1	0	0	0	0	0	92	91	92	90
NCVXQP2	0	0	0	0	0	148	217	156	216
NCVXQP3	19	14	19	17	4	228	293	237	215
NCVXQP4	0	0	0	0	0	11	15	11	14
NCVXQP5	0	0	0	0	0	6	7	6	6
NCVXQP6	53	50	53	53	3	11	133	10	13

Table 8.5: Number of superbasics and factorizations for CUTEr problems (continued)

	nS					nFac			
Name	lusol	ma57	umfpack	ma57v	sqopt	lusol	ma57	umfpack	ma57v
NCVXQP7	0	0	0	0	0	129	127	155	126
NCVXQP8	0	0	0	0	0	167	188	169	187
NCVXQP9	6	6	6	6	1	196	225	197	224
PORTSNQP	80	80	80	80	80	13	14	13	13
PORTSQP	99	99	99	99	99	17	14	17	17
POWELL20	1	1	1	1	1	10	6	10	10
PRIMAL1	133	262	133	130	131	4	1	4	4
PRIMAL2	302	557	302	302	300	5	1	5	5
PRIMAL3	572	648	572	572	570	5	1	5	5
PRIMAL4	1140	1427	1140	1140	1140	7	1	7	7
PRIMALC1	14	14	14	14	14	3	2	3	3
PRIMALC2	1	1	1	1	1	1	1	1	1
PRIMALC5	5	5	5	5	5	2	1	2	2
PRIMALC8	17	17	17	17	17	4	2	4	4
QPBAND	39	39	39	39	39	21	21	21	21
QPCBLEND	0	2	2	2	2	8	23	23	23
QPCBOEI1	111	111	111	111	108	106	107	108	107
QPCBOEI2	37	37	37	37	37	27	30	31	30
QPCSTAIR	0	33	34	33	27	19	42	42	42
QPNBAND	1	1	1	1	0	11	11	11	11
QPNBLEND	0	3	3	3	1	8	21	21	21
QPNBOEI1	93	93	93	93	22	74	76	66	76
QPNBOEI2	31	31	31	31	12	20	22	22	22
QPNSTAIR	0	31	31	31	25	41	66	68	66
S268	4	5	5	5	5	2	1	2	2
SOSQP1	0	9999	0	0	0	1	1	1	1
SOSQP2	4976	4985	4976	4976	4979	10	15	10	10
STATIC3	1	198	1	1	4	1	2	1	1
STCQP1	5707	5717	5707	5707	5707	6	1	6	6
STCQP2	3970	3970	3970	3970	3970	4	1	4	4
STEENBRA	11	11	11	11	11	1	1	1	1
STNQP1	5277	5277	5277	5277	0	6	56	6	6
STNQP2	2640	2640	2640	2640	0	3	381	3	3
TAME	1	1	1	1	1	3	3	3	3
UBH1	31	5997	5997	5997	471	47	6	1872	2092
WALL10	0	1056	0	0	0	2	0	2	2
WALL100	0	105074	0	0	0	2	0	2	2
WALL20	0	4214	0	0	0	2	0	2	2
WALL50	0	26286	0	0	0	2	0	2	2
YAO	1	1	1	1	1	1	1	1	1
ZECEVIC2	1	1	1	1	1	4	4	4	4

Table 8.6: Number of superbasics and factorizations for Maros and Mészáros problems

	nS					nFac			
Name	lusol	ma57	umfpack	ma57v	sqopt	lusol	ma57	umfpack	ma57v
ADAT1	1	1	1	1	1	1	1	1	1
ADAT2	6	6	6	6	6	2	2	2	2
ADAT3	6	6	6	6	6	2	1	2	2
AUG2D	10192	10196	10192	10192	10192	11	1	11	11
AUG2DC	10200	10200	10200	10200	10200	11	1	11	11
AUG2DCQP	9994	9994	9994	9994	9994	18	18	18	18
AUG2DQP	9801	9801	9801	9801	9801	18	18	17	18
AUG3D	2158	2158	2158	2158	2161	3	1	3	3
AUG3DC	2873	2873	2873	2873	2873	3	1	3	3
AUG3DCQP	2333	2333	2333	2333	2333	3	3	3	3
AUG3DQP	1455	1455	1455	1455	1455	2	2	2	2
CONT-050	195	195	195	195	195	1	1	1	1
CONT-100	395	395	395	395	395	3	3	3	3
CONT-101	97	97	97	97	97	1	1	1	1
CONT-200	795	795	795	795	795	1	1	1	1
CONT-201	197	197	197	197	197	1	1	1	1
CONT-300	297	297	297	297	297	7	1	7	1
CVXQP1_L	0	0	0	0	1275	1	3	272	3
CVXQP1_M	118	118	118	118	118	73	76	72	76
CVXQP1_S	14	14	14	14	14	1	1	1	1
CVXQP2_L	615	2210	2210	2210	2210	129	280	264	280
CVXQP2_M	217	217	217	217	217	9	9	10	9
CVXQP2_S	21	21	21	21	21	2	2	2	2
CVXQP3_L	0	0	1	0	434	3	3	663	3
CVXQP3_M	41	41	41	41	41	93	91	81	91
CVXQP3_S	3	3	3	3	3	1	1	1	1
DPKLO1	56	56	56	56	56	1	1	1	1
DTOC3	4803	4999	4803	4803	4803	5	1	5	5
DUAL1	62	62	62	62	62	14	14	14	14
DUAL2	91	91	91	91	91	15	15	15	15
DUAL3	96	96	96	96	96	15	15	15	15
DUAL4	61	61	61	61	61	13	13	13	13
DUALC1	2	2	2	2	2	1	1	1	1
DUALC2	2	2	2	2	2	1	1	1	1
DUALC5	4	4	4	4	4	1	1	1	1
DUALC8	2	2	2	2	2	1	1	1	1
EXDATA	421	421	421	421	421	4	3	4	4
GENHS28	2	2	2	2	2	1	1	1	1
GOULDQP2	306	306	306	306	305	1	1	1	1
GOULDQP3	174	174	174	174	174	1	1	1	1
HS118	0	0	0	0	0	1	1	1	1
HS21	0	1	0	0	0	2	2	2	2

Table 8.6: Number of superbasics and factorizations for Maros and Mészáros problems (continued)

	nS					nFac			
Name	lusol	ma57	umfpack	ma57v	sqopt	lusol	ma57	umfpack	ma57v
HS268	5	5	5	5	5	2	1	2	2
HS35	2	2	2	2	2	6	6	6	6
HS35MOD	2	2	2	2	2	4	4	4	4
HS51	2	2	2	2	2	1	1	1	1
HS52	2	2	2	2	2	1	1	1	1
HS53	2	2	2	2	2	1	1	1	1
HS76	2	2	2	2	2	2	2	2	2
HUES-MOD	8322	8322	8322	8322	8324	27	27	27	27
HUESTIS	3	8322	8322	8322	9146	4	27	27	27
KSIP	1	18	18	18	18	236	304	303	306
LASER	70	70	70	70	70	1	1	1	1
LISWET1	2	2	2	2	2	2	1	2	2
LISWET10	14	17	14	17	15	12	3	12	12
LISWET11	31	36	31	31	31	8	7	8	8
LISWET12	5	6	5	5	5	10	7	10	10
LISWET2	4	4	4	4	13	2	3	2	2
LISWET3	261	261	261	261	283	7	11	7	8
LISWET4	269	269	269	269	284	8	13	8	7
LISWET5	254	254	254	254	268	6	10	6	6
LISWET6	222	222	222	231	238	10	15	10	21
LISWET7	2	2	2	2	2	2	1	2	2
LISWET8	13	14	13	13	15	6	3	6	6
LISWET9	4	5	4	4	4	10	5	10	11
LOTSCHD	0	0	0	0	0	1	1	1	1
MOSARQP1	897	1012	1012	1012	1012	47	1716	1670	1716
MOSARQP2	273	568	568	568	568	23	29	29	29
POWELL20	1	1	1	1	1	10	6	10	10
PRIMAL1	133	262	133	130	131	4	1	4	4
PRIMAL2	302	557	302	302	300	5	1	5	5
PRIMAL3	572	648	572	572	570	5	1	5	5
PRIMAL4	1140	1427	1140	1140	1140	7	1	7	7
PRIMALC1	14	14	14	14	14	3	2	3	3
PRIMALC2	1	1	1	1	1	1	1	1	1
PRIMALC5	5	5	5	5	5	2	1	2	2
PRIMALC8	17	17	17	17	17	4	2	4	4
Q25FV47	1	2	37	2	38	672	876	1058	876
QADLITTL	6	6	6	6	6	8	8	8	8
QAFIRO	1	1	1	1	1	1	1	1	1
QBANDM	2	2	2	2	2	18	18	18	18
QBEACONF	0	0	0	0	0	1	1	1	1
QBORE3D	0	0	0	0	0	1	1	1	1
QBRANDY	5	5	5	5	5	16	16	16	16
QCAPRI	0	0	0	0	0	19	24	19	24

Table 8.6: Number of superbasics and factorizations for Maros and Mészáros problems (continued)

	nS					nFac			
Name	lusol	ma57	umfpack	ma57v	sqopt	lusol	ma57	umfpack	ma57v
QE226	37	37	37	37	27	43	23	43	23
QETAMACR	81	81	81	81	81	54	53	48	53
QFFFFF80	50	50	50	50	35	1	1	1	1
QFORPLAN	10	10	10	10	10	19	24	19	24
QGFRDXPN	5	5	5	5	4	62	64	62	64
QGROW15	1	1	1	1	1	12	11	10	11
QGROW22	9	9	9	9	1	91	131	138	131
QGROW7	1	1	1	1	1	8	16	8	16
QISRAEL	4	4	4	4	1	4	5	4	5
QPCBLEND	0	2	2	2	2	8	23	23	23
QPCBOEI1	111	111	111	111	111	93	94	93	94
QPCBOEI2	37	37	37	37	37	17	17	17	17
QPCSTAIR	35	33	34	33	31	49	53	52	53
QPILOTNO	1	0	2	0	2	58	2	325	2
QPTEST	1	1	1	1	1	1	1	1	1
QRECIPE	0	0	0	0	0	1	1	1	1
QSC205	1	1	1	1	1	1	1	1	1
QSCAGR25	4	4	4	4	1	18	18	18	18
QSCAGR7	1	1	1	1	1	1	1	1	1
QSCFXM1	20	20	20	20	20	13	13	13	13
QSCFXM2	20	20	20	20	19	8	8	8	8
QSCFXM3	22	22	22	22	22	12	12	12	12
QSCORPIO	0	0	0	0	0	2	2	2	2
QSCRS8	0	0	0	0	?	1	1	1	1
QSCSD1	0	0	0	0	0	3	3	4	3
QSCSD6	4	4	4	4	1	4	7	5	7
QSCSD8	16	17	16	17	4	36	25	51	25
QSCTAP1	0	0	0	0	0	1	1	1	1
QSCTAP2	5	5	5	5	5	7	7	7	7
QSCTAP3	11	11	11	11	4	14	8	8	8
QSEBA	13	13	13	13	14	1	1	1	1
QSHARE1B	10	10	10	10	8	1	1	1	1
QSHARE2B	0	0	0	0	0	1	1	1	1
QSHELL	62	62	62	62	18	1	1	1	1
QSHIP04L	3	3	3	3	3	2	2	2	2
QSHIP04S	3	3	3	3	3	1	1	1	1
QSHIP08L	19	19	19	19	9	5	5	5	5
QSHIP08S	15	15	15	15	6	6	6	6	6
QSHIP12L	43	43	43	43	4	19	20	19	20
QSHIP12S	44	44	44	44	17	2	2	2	2
QSIERRA	16	16	16	16	13	1	1	1	1
QSTAIR	1	1	1	1	1	19	12	19	12
QSTANDAT	18	18	18	18	18	1	1	1	1

Table 8.6: Number of superbasics and factorizations for Maros and Mészáros problems (continued)

	nS					nFac			
Name	lusol	ma57	umfpack	ma57v	sqopt	lusol	ma57	umfpack	ma57v
S268	5	5	5	5	5	2	1	2	2
STCQP1	225	2812	225	225	225	1	1	1	1
STCQP2	658	1940	658	658	658	1	1	1	1
TAME	1	1	1	1	1	3	3	3	3
UBH1	31	5997	5997	5997	471	47	6	1872	2092
VALUES	23	23	23	23	23	16	16	16	16
YAO	1	1	1	1	1	1	1	1	1
ZECEVIC2	1	1	1	1	1	4	4	4	4

Table 8.7: Number of temporarily fixed variables for a vertex for CUTEr problems

Name	nFix	Name	nFix	Name	nFix	Name	nFix
ALLINQP	1428	DUAL4	0	HS76I	4	NCVXQP9	199
AUG2D	10200	DUALC1	0	HUES-MOD	9995	PORTSNQP	1
AUG2DC	10200	DUALC2	0	HUESTIS	9995	PORTSQP	2
AUG2DCQP	0	DUALC5	0	KSIP	20	POWELL20	4998
AUG2DQP	0	DUALC8	0	LINCONT	0	PRIMAL1	323
AUG3D	19543	FERRISDC	1	LISWET1	2	PRIMAL2	647
AUG3DC	19543	GENHS28	2	LISWET10	2	PRIMAL3	743
AUG3DCQP	0	GMNCASE1	111	LISWET11	2	PRIMAL4	1487
AUG3DQP	0	GMNCASE2	121	LISWET12	2	PRIMALC1	14
AVGASA	1	GMNCASE3	99	LISWET2	2	PRIMALC2	1
AVGASB	1	GMNCASE4	0	LISWET3	2	PRIMALC5	8
BIGGSC4	0	GOULDQP2	0	LISWET4	2	PRIMALC8	16
BLOCKQP1	5007	GOULDQP3	0	LISWET5	2	QPBAND	0
BLOCKQP2	5007	GRIDNETA	2081	LISWET6	2	QPCBLEND	0
BLOCKQP3	5007	GRIDNETB	6561	LISWET7	2	QPCBOEI1	0
BLOCKQP4	5007	GRIDNETC	2187	LISWET8	2	QPCBOEI2	0
BLOCKQP5	5003	HATFLDH	1	LISWET9	2	QPCSTAIR	0
BLOWEYA	2001	HS118	14	LOTSCHD	0	QPNBAND	0
BLOWEYB	2001	HS21	1	MOSARQP1	2487	QPNBLEND	0
BLOWEYC	2001	HS268	3	MOSARQP2	2487	QPNBOEI1	0
CONT5-QP	795	HS35	3	NASH	0	QPNBOEI2	0
CVXQP1	4366	HS35I	3	NCVXQP1	446	QPNSTAIR	0
CVXQP2	7325	HS35MOD	2	NCVXQP2	446	S268	3
CVXQP3	2006	HS44	0	NCVXQP3	446	SOSQP1	10000
DEGENQP	0	HS44NEW	4	NCVXQP4	731	SOSQP2	5008
DTOC3	4999	HS51	2	NCVXQP5	731	STATIC3	266
DUAL1	0	HS52	2	NCVXQP6	731	STCQP1	6422
DUAL2	0	HS53	2	NCVXQP7	199	STCQP2	4098
DUAL3	0	HS76	4	NCVXQP8	199	STEENBRA	0

Table 8.7: Number of temporarily fixed variables for a vertex for CUTEr problems (continued)

Name	nFix	Name	nFix	Name	nFix	Name	nFix
STNQP1	6422	UBH1	8340	WALL20	4214	ZECEVIC2	1
STNQP2	4098	WALL10	1056	WALL50	26286		
TAME	0	WALL100	105074	YAO	0		

Table 8.8: Number of temporarily fixed variables for a vertex for Maros and Mészáros problems

Name	nFix	Name	nFix	Name	nFix	Name	nFix
ADAT1	3	DUALC2	0	POWELL20	4998	QSCAGR7	0
ADAT2	3	DUALC5	0	PRIMAL1	323	QSCFXM1	0
ADAT3	3	DUALC8	0	PRIMAL2	647	QSCFXM2	0
AUG2D	10200	EXDATA	1499	PRIMAL3	743	QSCFXM3	0
AUG2DC	10200	GENHS28	2	PRIMAL4	1487	QSCORPIO	0
AUG2DCQP	0	GOULDQP2	0	PRIMALC1	14	QSCRS8	0
AUG2DQP	0	GOULDQP3	0	PRIMALC2	1	QSCSD1	0
AUG3D	2873	HS118	0	PRIMALC5	8	QSCSD6	0
AUG3DC	2873	HS21	1	PRIMALC8	16	QSCSD8	0
AUG3DCQP	0	HS268	4	Q25FV47	0	QSCTAP1	0
AUG3DQP	0	HS35	0	QADLITTL	0	QSCTAP2	0
BOYD1	?	HS35MOD	0	QAFIRO	0	QSCTAP3	0
BOYD2	?	HS51	2	QBANDM	0	QSEBA	0
CONT-050	0	HS52	2	QBEACONF	0	QSHARE1B	0
CONT-100	0	HS53	2	QBORE3D	0	QSHARE2B	0
CONT-101	0	HS76	0	QBRANDY	0	QSHELL	0
CONT-200	0	HUES-MOD	0	QCAPRI	0	QSHIP04L	0
CONT-201	0	HUESTIS	0	QE226	0	QSHIP04S	0
CONT-300	0	KSIP	18	QETAMACR	0	QSHIP08L	0
CVXQP1_L	0	LASER	2	QFFFFF80	0	QSHIP08S	0
CVXQP1_M	0	LISWET1	2	QFORPLAN	0	QSHIP12L	0
CVXQP1_S	0	LISWET10	2	QGFRDXPN	0	QSHIP12S	0
CVXQP2_L	0	LISWET11	2	QGROW15	0	QSIERRA	0
CVXQP2_M	0	LISWET12	2	QGROW22	0	QSTAIR	0
CVXQP2_S	0	LISWET2	2	QGROW7	0	QSTANDAT	0
CVXQP3_L	0	LISWET3	2	QISRAEL	0	S268	4
CVXQP3_M	0	LISWET4	2	QPCBLEND	0	STCQP1	3158
CVXQP3_S	0	LISWET5	2	QPCBOEI1	0	STCQP2	2045
DPKLO1	56	LISWET6	2	QPCBOEI2	0	TAME	0
DTOC3	4999	LISWET7	2	QPCSTAIR	0	UBH1	8340
DUAL1	0	LISWET8	2	QPILOTNO	0	VALUES	0
DUAL2	0	LISWET9	2	QPTEST	0	YAO	0
DUAL3	0	LOTSCHD	0	QRECIPE	0	ZECEVIC2	0
DUAL4	0	MOSARQP1	0	QSC205	0		
DUALC1	0	MOSARQP2	0	QSCAGR25	0		

A Test Problem Data

Table A.1: Problem sizes for CUTEr QPs

Name	m	n	Name	m	n	Name	m	n
ALLINQP	5000	10000	DUAL3	1	111	HS53	3	5
AUG2D	10000	20200	DUAL4	1	75	HS76	3	4
AUG2DC	10000	20200	DUALC1	215	9	HS76I	3	4
AUG2DCQP	10000	20200	DUALC2	229	7	HUES-MOD	2	10000
AUG2DQP	10000	20200	DUALC5	278	8	HUESTIS	2	10000
AUG3D	8000	27543	DUALC8	503	8	KSIP	1001	20
AUG3DC	8000	27543	FERRISDC^i	320	6300	LINCONT^i	419	1257
AUG3DCQP	8000	27543	GENHS28	8	10	LISWET1	10000	10002
AUG3DQP	8000	27543	GMNCASE1^i	300	175	LISWET10	10000	10002
AVGASA	10	8	GMNCASE2	1050	175	LISWET11	10000	10002
AVGASB	10	8	GMNCASE3	1050	175	LISWET12	10000	10002
BIGGSC4^i	7	4	GMNCASE4	350	175	LISWET2	10000	10002
BLOCKQP1^i	5001	10010	GOULDQP2	9999	19999	LISWET3	10000	10002
BLOCKQP2^i	5001	10010	GOULDQP3	9999	19999	LISWET4	10000	10002
BLOCKQP3^i	5001	10010	GRIDNETA	6724	13284	LISWET5	10000	10002
BLOCKQP4^i	5001	10010	GRIDNETB	6724	13284	LISWET6	10000	10002
BLOCKQP5^i	5001	10010	GRIDNETC	6724	13284	LISWET7	10000	10002
BLOWEYA^i	2002	4002	HATFLDH^i	7	4	LISWET8	10000	10002
BLOWEYB^i	2002	4002	HS118	17	15	LISWET9	10000	10002
BLOWEYC^i	2002	4002	HS21	1	2	LOTSCHD	7	12
CONT5-QP	40200	40601	HS268	5	5	MOSARQP1	700	2500
CVXQP1	5000	10000	HS35	1	3	MOSARQP2	700	2500
CVXQP2	2500	10000	HS35I	1	3	NASH^i	24	72
CVXQP3	7500	10000	HS35MOD	1	3	NCVXQP1^i	500	1000
DEGENQP	8010	20	HS44^i	6	4	NCVXQP2^i	500	1000
DTOC3	9998	14999	HS44NEW^i	6	4	NCVXQP3^i	500	1000
DUAL1	1	85	HS51	3	5	NCVXQP4^i	250	1000
DUAL2	1	96	HS52	3	5	NCVXQP5^i	250	1000

Table A.1: Problem sizes for CUTEr QPs (continued)

Name	m	n	Name	m	n	Name	m	n
NCVXQP6^i	250	1000	PRIMALC8	8	520	STATIC3^i	96	434
NCVXQP7^i	750	1000	QPBAND	5000	10000	STCQP1	4095	8193
NCVXQP8^i	750	1000	QPCBLEND	74	83	STCQP2	4095	8193
NCVXQP9^i	750	1000	QPCBOEI1	351	384	STEENBRA	108	432
PORTSNQP^i	2	10000	QPCBOEI2	166	143	STNQP1^i	4095	8193
PORTSQP	1	10000	QPCSTAIR	356	467	STNQP2^i	4095	8193
POWELL20	10000	10000	QPNBAND^i	5000	10000	TAME	1	2
PRIMAL1	85	325	QPNBLEND^i	74	83	UBH1	12000	18009
PRIMAL2	96	649	QPNBOEI1^i	351	384	WALL10	1	1461
PRIMAL3	111	745	QPNBOEI2^i	166	143	WALL20	1	5924
PRIMAL4	75	1489	QPNSTAIR^i	356	467	WALL50	1	37311
PRIMALC1	9	230	S268	5	5	WALL100	1	149624
PRIMALC2	7	231	SOSQP1^i	10001	20000	YAO	2000	2002
PRIMALC5	8	287	SOSQP2^i	10001	20000	ZECEVIC2	2	2

Table A.2: Problem sizes for Maros and Mészáros QPs

Name	m	n	Name	m	n	Name	m	n
AUG2D	10000	20200	CVXQP2_M	250	1000	HS21	1	2
AUG2DC	10000	20200	CVXQP2_S	25	100	HS268	5	5
AUG2DCQP	10000	20200	CVXQP3_L	7500	10000	HS35	1	3
AUG2DQP	10000	20200	CVXQP3_M	750	1000	HS35MOD	1	3
AUG3D	1000	3873	CVXQP3_S	75	100	HS51	3	5
AUG3DC	1000	3873	DPKLO1	77	133	HS52	3	5
AUG3DCQP	1000	3873	DTOC3	9998	14999	HS53	3	5
AUG3DQP	1000	3873	DUAL1	1	85	HS76	3	4
BOYD1	18	93261	DUAL2	1	96	HUES-MOD	2	10000
BOYD2	186531	93263	DUAL3	1	111	HUESTIS	2	10000
CONT-050	2401	2597	DUAL4	1	75	KSIP	1001	20
CONT-100	9801	10197	DUALC1	215	9	LASER	1000	1002
CONT-101	10098	10197	DUALC2	229	7	LISWET1	10000	10002
CONT-200	39601	40397	DUALC5	278	8	LISWET10	10000	10002
CONT-201	40198	40397	DUALC8	503	8	LISWET11	10000	10002
CONT-300	90298	90597	EXDATA	3001	3000	LISWET12	10000	10002
CVXQP1_L	5000	10000	GENHS28	8	10	LISWET2	10000	10002
CVXQP1_M	500	1000	GOULDQP2	349	699	LISWET3	10000	10002
CVXQP1_S	50	100	GOULDQP3	349	699	LISWET4	10000	10002
CVXQP2_L	2500	10000	HS118	17	15	LISWET5	10000	10002

Table A.2: Problem sizes for Maros and Mészáros QPs (continued)

Name	m	n	Name	m	n	Name	m	n
LISWET6	10000	10002	QFFFFF80	524	854	QSCTAP2	1090	1880
LISWET7	10000	10002	QFORPLAN	161	421	QSCTAP3	1480	2480
LISWET8	10000	10002	QGFRDXPN	616	1092	QSEBA	515	1028
LISWET9	10000	10002	QGROW15	300	645	QSHARE1B	117	225
LOTSCHD	7	12	QGROW22	440	946	QSHARE2B	96	79
MOSARQP1	700	2500	QGROW7	140	301	QSHELL	536	1775
MOSARQP2	600	900	QISRAEL	174	142	QSHIP04L	402	2118
POWELL20	10000	10000	QPCBLEND	74	83	QSHIP04S	402	1458
PRIMAL1	85	325	QPCBOEI1	351	384	QSHIP08L	778	4283
PRIMAL2	96	649	QPCBOEI2	166	143	QSHIP08S	778	2387
PRIMAL3	111	745	QPCSTAIR	356	467	QSHIP12L	1151	5427
PRIMAL4	75	1489	QPILOTNO	975	2172	QSHIP12S	1151	2763
PRIMALC1	9	230	QPTEST	2	2	QSIERRA	1227	2036
PRIMALC2	7	231	QRECIPE	91	180	QSTAIR	356	467
PRIMALC5	8	287	QSC205	205	203	QSTANDAT	359	1075
PRIMALC8	8	520	QSCAGR25	471	500	S268	5	5
Q25FV47	820	1571	QSCAGR7	129	140	STADAT1	3999	2001
QADLITTL	56	97	QSCFXM1	330	457	STADAT2	3999	2001
QAFIRO	27	32	QSCFXM2	660	914	STADAT3	7999	4001
QBANDM	305	472	QSCFXM3	990	1371	STCQP1	2052	4097
QBEACONF	173	262	QSCORPIO	388	358	STCQP2	2052	4097
QBORE3D	233	315	QSCRS8	490	1169	TAME	1	2
QBRANDY	220	249	QSCSD1	77	760	UBH1	12000	18009
QCAPRI	271	353	QSCSD6	147	1350	VALUES	1	202
QE226	223	282	QSCSD8	397	2750	YAO	2000	2002
QETAMACR	400	688	QSCTAP1	300	480	ZECEVIC2	2	2

Bibliography

[1] A. Altman and J. Gondzio. Regularized symmetric indefinite systems in interior point methods for linear and quadratic optimization. *Optim. Methods Softw.*, 11/12(1-4):275–302, 1999.

[2] R. A. Bartlett and L. T. Biegler. QPSchur: a dual, active-set, Schur-complement method for large-scale and structured convex quadratic programming. *Optim. Eng.*, 7(1):5–32, 2006.

[3] M. C. Biggs. Constrained minimization using recursive equality quadratic programming. In F. A. Lootsma, editor, *Numerical Methods for Nonlinear Optimization*, pages 411–428. Academic Press, London and New York, 1972.

[4] J. Bisschop and A. Meeraus. Matrix augmentation and partitioning in the updating of the basis inverse. *Math. Program.*, 13:241–254, 1977.

[5] N. L. Boland. A dual-active-set algorithm for positive semi-definite quadratic programming. *Math. Programming*, 78(1, Ser. A):1–27, 1997.

[6] I. Bongartz, A. R. Conn, N. I. M. Gould, and P. L. Toint. CUTE: Constrained and unconstrained testing environment. *ACM Trans. Math. Software*, 21(1):123–160, 1995.

[7] J. M. Borwein. Necessary and sufficient conditions for quadratic minimality. *Numer. Funct. Anal. and Optimiz.*, 5:127–140, 1982.

[8] G. G. Brown and G. W. Graves. Elastic programming: A new approach to large-scale mixed-integer optimization, November 1975. Presented at the presented at ORSA/TIMS meeting, Las Vegas, NV.

[9] A. R. Conn, N. I. M. Gould, and P. L. Toint. Global convergence of a class of trust region algorithms for optimization with simple bounds. *SIAM J. Numer. Anal.*, 25:433–460, 1988.

[10] A. R. Conn, N. I. M. Gould, and P. L. Toint. A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. *SIAM J. Numer. Anal.*, 28:545–572, 1991.

[11] A. R. Conn, N. I. M. Gould, and P. L. Toint. `LANCELOT`*: a Fortran package for large-scale nonlinear optimization (Release A)*. Lecture Notes in Computation Mathematics 17. Springer Verlag, Berlin, Heidelberg, New York, London, Paris and Tokyo, 1992.

[12] L. B. Contesse. Une caractérisation complète des minima locaux en programmation quadratique. *Numer. Math.*, 34:315–332, 1980.

[13] COPS. Constrained optimization problem set. `http://www.mcs.anl.gov/~more/cops`.

[14] R. W. Cottle, G. J. Habetler, and C. E. Lemke. On classes of copositive matrices. *Linear Algebra Appl.*, 3:295–310, 1970.

[15] T. A. Davis. Algorithm 832: UMFPACK V4.3—an unsymmetric-pattern multifrontal method. *ACM Trans. Math. Software*, 30(2):196–199, 2004.

[16] T. A. Davis. A column pre-ordering strategy for the unsymmetric-pattern multifrontal method. *ACM Trans. Math. Software*, 30(2):167–195, 2004.

[17] T. A. Davis and I. S. Duff. An unsymmetric-pattern multifrontal method for sparse *LU* factorization. *SIAM J. Matrix Anal. Appl.*, 18(1):140–158, 1997.

[18] T. A. Davis and I. S. Duff. A combined unifrontal/multifrontal method for unsymmetric sparse matrices. *ACM Trans. Math. Software*, 25(1):1–20, 1999.

[19] F. Delbos and J. C. Gilbert. Global linear convergence of an augmented Lagrangian algorithm to solve convex quadratic optimization problems. *J. Convex Anal.*, 12(1):45–69, 2005.

[20] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Math. Program.*, 91(2, Ser. A):201–213, 2002.

[21] W. S. Dorn. Duality in quadratic programming. *Quart. Appl. Math.*, 18:155–162, 1960/1961.

[22] Z. Dostál, A. Friedlander, and S. A. Santos. Adaptive precision control in quadratic programming with simple bounds and/or equalities. In *High performance algorithms and software in nonlinear optimization (Ischia, 1997)*, volume 24 of *Appl. Optim.*, pages 161–173. Kluwer Acad. Publ., Dordrecht, 1998.

[23] Z. Dostál, A. Friedlander, and S. A. Santos. Augmented Lagrangians with adaptive precision control for quadratic programming with equality constraints. *Comput. Optim. Appl.*, 14(1):37–53, 1999.

[24] Z. Dostál, A. Friedlander, and S. A. Santos. Augmented Lagrangians with adaptive precision control for quadratic programming with simple bounds and equality constraints. *SIAM J. Optim.*, 13(4):1120–1140 (electronic), 2003.

[25] I. S. Duff. MA57—a code for the solution of sparse symmetric definite and indefinite systems. *ACM Trans. Math. Software*, 30(2):118–144, 2004.

[26] I. S. Duff. Subroutine `ma57lfd.f`, October 2010. Private communication.

[27] S. K. Eldersveld and M. A. Saunders. A block-LU update for large-scale linear programming. *SIAM J. Matrix Anal. Appl.*, 13:191–201, 1992.

[28] A. V. Fiacco and G. P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques.* John Wiley and Sons, Inc., New York-London-Sydney, 1968.

[29] R. Fletcher. A general quadratic programming algorithm. *J. Inst. Math. Applics.*, 7:76–91, 1971.

[30] R. Fletcher. An ℓ_1 penalty method for nonlinear constraints. In P. T. Boggs, R. H. Byrd, and R. B. Schnabel, editors, *Numerical Optimization 1984*, pages 26–40, Philadelphia, 1985.

[31] R. Fletcher and S. Leyffer. User manual for filterSQP. Technical Report NA/181, Dept. of Mathematics, University of Dundee, Scotland, 1998.

[32] A. Forsgren, P. E. Gill, and W. Murray. On the identification of local minimizers in inertia-controlling methods for quadratic programming. *SIAM J. Matrix Anal. Appl.*, 12:730–746, 1991.

[33] M. P. Friedlander. *A Globally Convergent Linearly Constrained Lagrangian Method for Nonlinear Optimization.* PhD thesis, Department of Operations Research, Stanford University, Stanford, CA, 2002.

[34] M. P. Friedlander and S. Leyffer. Global and finite termination of a two-phase augmented Lagrangian filter method for general quadratic programs. *SIAM J. Sci. Comput.*, 30(4):1706–1729, 2008.

[35] M. P. Friedlander and M. A. Saunders. A globally convergent linearly constrained Lagrangian method for nonlinear optimization. *SIAM J. Optim.*, 15(3):863–897, 2005.

[36] P. E. Gill, W. Murray, D. B. Ponceleón, and M. A. Saunders. Solving reduced KKT systems in barrier methods for linear programming. In G. A. Watson and D. Griffiths, editors, *Numerical Analysis 1993 (Dundee, 1993)*, volume 303 of *Pitman Res. Notes Math. Ser.*, pages 89–104. Longman Sci. Tech., Harlow, UK, 1994.

[37] P. E. Gill, W. Murray, and M. A. Saunders. User's guide for QPOPT 1.0: a Fortran package for quadratic programming. Report SOL 95-4, Department of Operations Research, Stanford University, Stanford, CA, 1995.

[38] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Rev.*, 47:99–131, 2005.

[39] P. E. Gill, W. Murray, and M. A. Saunders. User's guide for SQOPT Version 7: Software for large-scale linear and quadratic programming. Numerical Analysis Report 06-1, Department of Mathematics, University of California, San Diego, La Jolla, CA, 2006.

[40] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. Sparse matrix methods in optimization. *SIAM J. Sci. Statist. Comput.*, 5(3):562–589, 1984.

[41] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. Maintaining *LU* factors of a general sparse matrix. *Linear Algebra Appl.*, 88/89:239–270, 1987.

[42] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. A Schur-complement method for sparse quadratic programming. In M. G. Cox and S. J. Hammarling, editors, *Reliable Numerical Computation*, pages 113–138. Oxford University Press, 1990.

[43] P. E. Gill and D. P. Robinson. A primal-dual augmented Lagrangian. *Computational Optimization and Applications*, pages 1–25, 2010.

[44] P. E. Gill and E. Wong. Sequential quadratic programming methods. In J. Lee and S. Leyffer, editors, *Mixed-Integer Nonlinear Optimization: Algorithmic Advances and Applications*, The IMA Volumes in Mathematics and its Applications. Springer Verlag, Berlin, Heidelberg and New York, 2011.

[45] P. E. Gill and E. Wong. User's guide for icQP. Numerical Analysis Report 11-3, Department of Mathematics, University of California, San Diego, La Jolla, CA, 2011.

[46] P. E. Gill and E. Wong. User's guide for SNCTRL, an interface for SNOPT 7. Numerical Analysis Report 11-1, Department of Mathematics, University of California, San Diego, La Jolla, CA, 2011.

[47] D. Goldfarb and A. Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Math. Programming*, 27(1):1–33, 1983.

[48] N. I. M. Gould. On the accurate determination of search directions for simple differentiable penalty functions. *IMA J. Numer. Anal.*, 6:357–372, 1986.

[49] N. I. M. Gould. An algorithm for large-scale quadratic programming. *IMA J. Numer. Anal.*, 11(3):299–324, 1991.

[50] N. I. M. Gould, D. Orban, and P. L. Toint. CUTEr and SifDec: A constrained and unconstrained testing environment, revisited. *ACM Trans. Math. Software*, 29(4):373–394, 2003.

[51] N. I. M. Gould, D. Orban, and P. L. Toint. *GALAHAD*, a library of thread-safe Fortran 90 packages for large-scale nonlinear optimization. *ACM Trans. Math. Software*, 29(4):353–372, Dec 2003.

[52] N. I. M. Gould, J. A. Scott, and Y. Hu. A numerical evaluation of sparse direct solvers for the solution of large sparse symmetric linear systems of equations. *ACM Trans. Math. Software*, 33(2):Art. 10, 32, 2007.

[53] H. M. Huynh. *A Large-Scale Quadratic Programming Solver Based on Block-LU Updates of the KKT System*. PhD thesis, Program in Scientific Computing and Computational Mathematics, Stanford University, Stanford, CA, 2008.

[54] C. M. Maes. *A Regularized Active-Set Method for Sparse Convex Quadratic Programming*. PhD thesis, Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA, August 2010.

[55] A. Majthay. Optimality conditions for quadratic programming. *Math. Programming*, 1:359–365, 1971.

[56] I. Maros and C. Mészáros. A repository of convex quadratic programming problems. *Optim. Methods Softw.*, 11/12(1-4):671–681, 1999. Interior point methods.

[57] W. Murray. An algorithm for constrained minimization. In *Optimization (Sympos., Univ. Keele, Keele, 1968)*, pages 247–258. Academic Press, London, 1969.

[58] P. M. Pardalos and G. Schnitger. Checking local optimality in constrained quadratic programming is NP-hard. *Oper. Res. Lett.*, 7(1):33–35, 1988.

[59] P. M. Pardalos and S. A. Vavasis. Quadratic programming with one negative eigenvalue is NP-hard. *J. Global Optim.*, 1(1):15–22, 1991.

[60] M. J. D. Powell. On the quadratic programming algorithm of Goldfarb and Idnani. *Math. Programming Stud.*, (25):46–61, 1985.

[61] D. P. Robinson. *Primal-Dual Methods for Nonlinear Optimization*. PhD thesis, Department of Mathematics, University of California, San Diego, September 2007.

[62] M. A. Saunders. LUMOD: Updating a dense square factorization LC = U. `http://www.stanford.edu/group/SOL/software/lumod.html`.

[63] M. A. Saunders. Cholesky-based methods for sparse least squares: The benefits of regularization. In L. Adams and J. L. Nazareth, editors, *Linear and Nonlinear Conjugate Gradient-Related Methods*, pages 92–100. SIAM Publications, Philadelphia, 1996. Proceedings of AMS-IMS-SIAM Joint Summer Research Conference, University of Washington, Seattle, WA (July 9–13, 1995).

[64] M. A. Saunders and J. A. Tomlin. Solving regularized linear programs using barrier methods and KKT systems. Report SOL 96-4, Dept of EESOR, Stanford University, 1996.

[65] M. A. Saunders and J. A. Tomlin. Stable reduction to KKT systems in barrier methods for linear and quadratic programming. Report SOL 96-3, Dept of EESOR, Stanford University, 1996.

[66] R. A. Tapia. Diagonalized multiplier methods and quasi-Newton methods for constrained optimization. *J. Optim. Theory Appl.*, 22:135–194, 1977.

[67] R. J. Vanderbei. LOQO: an interior-point code for quadratic programming. *Optimization Methods and Software*, 11(1–4):451–484, 1999.

[68] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.*, 106(1, Ser. A):25–57, 2006.