UNIVERSITÀ DEGLI STUDI DI NAPOLI "FEDERICO II"

Facoltà di Scienze Matematiche,
Fisiche e Naturali

Sonia Cafieri

On the application of iterative solvers to KKT systems in Interior Point methods for Large-Scale Quadratic Programming problems

Tesi di Dottorato di Ricerca in
Scienze Matematiche XVII ciclo

Tutore	Coordinatore
Ch.mo Prof. Marco D'Apuzzo	Ch.mo Prof. Salvatore Rionero

Preface

Interior Point methods for linear and nonlinear optimization problems have received an increasing attention in the last years. Main reasons for the interest in Interior Point methods are their very attractive computational efficiency and good theoretical convergence properties, and their applicability to several classes of nonlinear programming, semidefinite programming and complementarity problems. Such research interest is also demonstrated by the development of several software packages based on Interior Point algorithms.

A crucial issue in the development of efficient Interior Point software is the solution of the linear system, named KKT system, that arises at each iteration of the method. Sparse direct methods for linear systems are widely used in Interior Point based software, but when dealing with large-scale problems their computational cost may become prohibitive. A promising alternative is provided by iterative methods, that are receiving an increasing attention by optimization community. It is known that great care must be taken in the use of iterative methods for the linear systems. Firstly, since iterative methods just give an approximate solution of the linear systems, some of the Interior Point convergence theory have to be reassessed. Secondly, the success of an iterative approach depends on the use of a suitable preconditioner, that is mandatory because of the increasing ill-conditioning of the systems when the iterates generated by the Interior Point method approach the solution. Ad hoc preconditioners can be developed, that are able to tackle ill-conditioning. Furthermore, iterative solvers allow to use adaptive accuracy requirements in the solution of the systems to avoid unnecessary iterations when the current Interior Point iterate is far from the solution.

The aim of the research activity described in this thesis has been the analysis, the development and the implementation of iterative methods for

the efficient solution of the KKT systems arising at each iteration of Interior Point methods for large-scale convex Quadratic Programming problems.

We focus on KKT systems reduced to the well known augmented system and normal equations forms and we consider a preconditioned Conjugate Gradient method for their solution. Specifically we consider an incomplete Cholesky factorization with limited memory for the normal equations approach and a constraint preconditioner for the augmented system approach. In particular, in the last case we analyze the behaviour of the constraint preconditioner with the Conjugate Gradient algorithm and we prove, for KKT systems deriving from linear inequality constraints and nonnegativity bounds on the variables, the equivalence with a suitable preconditioned Conjugate Gradient applied to the positive definite normal equations. Starting from this equivalence, we prove that no breakdown occurs for the considered algorithm and it converges even if the augmented system matrix is indefinite.

The Interior Point framework is given by the Potential Reduction method. Such method has good convergence and complexity properties, furthermore it provides a quite simple framework to study linear algebra kernels that also arise in other more general contexts. We extend convergence results for the Potential Reduction method in the case of an iterative solution of the KKT system arising at each iteration, starting from convergence results in the case of direct solution of the system.

We have developed two software packages for solving large-scale convex quadratic problems, based on the considered methods. We discuss some implementation issues, with emphasis on those related to the solution of the KKT systems with iterative methods. In particular, we describe a computational study of stopping criteria of the preconditioned Conjugate Gradient method for solving the KKT systems. We present results of extensive numerical experiments carried out in oder to verify the effectiveness of the proposed approaches on a set of large-scale quadratic problems. We also compare our

software packages to a well-estabilished software for nonlinear optimization problems.

The thesis is organized as follows.

In Chapter 1 we outline the context of the research activity described in this thesis. After a description of the considered optimization problem, we focus on Interior Point methods for its solution. We describe the basic idea and we give a brief overview on main Interior Point approches. We present the KKT system arising at each iteration, whose efficient solution will be the subject of the following chapter.

Chapter 2 focuses on the analysis and the development of suitable iterative methods for solving the KKT systems. After a description of motivations leading to the choice of iterative approaches, mainly for large-scale problems, we analyze the solution of the normal equations and the augmented system with a suitable preconditioned Conjugate Gradient method. In the case of normal equations we consider an incomplete Cholesky factorization, while for the augmented system we consider a constraint preconditioner. We prove the equivalence between the Conjugate Gradient algorithm with the constraint preconditioner and the Conjugate Gradient algorithm with a suitable preconditioner applied to the normal equations, so that the Conjugate Gradient can be applied to the augmented system which has an indefinite matrix.

In Chapter 3 we describe the Potential Reduction method, which is the considered Interior Point framework for the Linear Algebra methods that we developed. We describe the basic idea and its main features. Our interest on the iterative solution of the linear system arising at each iteration leads to analyze the convergence properties in the case of iterative solution of the KKT systems. We extend to this case convergence results of the Potential Reduction method for Quadratic Programming in the case of direct solution of the KKT systems.

The Potential Reduction with iterative solution of KKT systems, based on an incomplete Cholesky factorization for the normal equations approach and on the constraint preconditioner for the augmented system approach, has been implemented in two software packages. The first one solves bound constrained quadratic problems, the second one solves quadratic problems with more general linear constraints. In Chapter 4 we describe our software packages and, to this aim, we discuss some implementation issues, with a particular attention for those related to the use of iterative methods for KKT systems. In this context, we present a computational study of stopping criteria for the preconditioned Conjugate Gradient, which we rely to the convergence properties of the Potential Reduction method. We show that a strategy based on restarting Conjugate Gradient iterations is particularly advantageous.

We report results of numerical experiments on a set of test problems with different sizes and sparsity patterns of the matrices. In the case of bound constrained problems we compare some available preconditioners for symmetric positive definite systems; in the case of problems with more general constraints, where the constraint preconditioner is the most promising preconditioner for the augmented system, we compare the iterative approach to a direct one. In both cases, we compare our solvers with the well-estabilished software MOSEK for nonlinear optimization.

Acknowledgments

I would thank my advisor Prof. Marco D'Apuzzo, who has believed in my research skills and has given me encouragement, guidance and support during the four years of my research activity. I learned a lot from my work with him.

I am very grateful to Prof. Daniela di Serafino for her constant advice and for having shared with me her knowledge and expertise on numerical Linear Algebra. Her suggestions and comments on this thesis have been very helpful.

I also wish to thank Dr. Valentina De Simone. Her competence, enthusiasm and positive energy always made work a great pleasure.

A special thank to Prof. Gerardo Toraldo, with whom I have had the privilege of working, for fruitful discussions about optimization, for his helpful comments and his interesting suggestions, that have always stimulated my work.

Finally, thanks to Antonio Mucherino for his love and support during all these years.

The work described in this thesis was partially supported by the Italian MIUR FIRB Project *Large Scale Nonlinear Optimization*, grant no. RBNE01WBBB.

Contents

Chapter 1

Interior Point methods for Quadratic Programming

1.1 Introduction

In this Chapter we outline the context within which the thesis has been developed. We state the target optimization problem, that is convex Quadratic Programming, and we introduce its primal-dual formulation and its optimality conditions.

Then we outline the basic idea and the most important approaches of Interior Point methods for solving the target problem. We describe how primal-dual Interior Point methods generate the sequence of iterates, how they compute at each iteration a search direction and how they move along such direction to obtain the next approximation of the solution.

We introduce the main topic of this thesis, that is the efficient solution of the linear system arising at each iteration of Interior Point methods, which is the subject of the next chapter.

We also discuss about the main differences between Interior Point methods and methods based on Active Set strategies.

Finally, we outline some of the currently available optimization software packages based on Interior Point algorithms.

1.2 Problem statement

1.2.1 The QP Problem

Convex Quadratic Programming (QP) problems, i.e. optimization problems with convex quadratic objective function and linear constraints, arise often as subproblems in general nonlinear constrained optimization. Methods based on *Sequential Quadratic Programming* use at each iteration a quadratic approximation of the problem in order to compute a search direction. Furthermore, there are a large number of other interesting applications of the QP problem, such as, for example, the *portfolio* model proposed by Markowitz [64], optimal control [10, 102] and VLSI design [103, 56].

We are particularly interested in solving large-scale QP problems. Mathematical modelization of phisical problems often leads to the formulation of optimization problems with tens or hundreds of thousands of variables. The increased power of computers and the development of algorithms which are able to take advantage of this power make possible to solve very large problems, so *large-scale* nonlinear optimization has become the most active research field in nonlinear optimization.

In the sequel we define the QP problem and we give some basic definitions and results. For more details the reader is referred to, e.g., [25, 35, 79].

We consider a convex QP problem in the following form:

$$\begin{array}{ll} \text{minimize} & q(x) = \frac{1}{2}x^TQx + c^Tx \\ \text{subject to} & Ax \geq b, \quad x \geq 0 \end{array} \tag{1.1}$$

where $Q \in \Re^{n\times n}$ is symmetric positive semidefinite, $A \in \Re^{m\times n}$, with $m \leq n$, $c, x \in \Re^n$, $b \in \Re^m$ and the inequalities are interpreted componentwise.
We assume that A has full rank m, i.e. all the constraints are linearly independent. If this is not the case, the problem has redundant constraints that can be removed, so that our assumption is without loss of generality.

The original problem (1.1) can be transformed, by introducing a *slack* vector $z \in \Re^m$, to the following form, usually referred as the *primal problem*:

$$\mathcal{P} \equiv \begin{cases} \text{minimize} & p(x) = \frac{1}{2}x^TQx + c^Tx \\ \text{subject to} & Ax - b = z, \quad x \geq 0, \quad z \geq 0 \end{cases} \tag{1.2}$$

The so-called *dual problem* can be associated to the primal one (1.2):

$$\mathcal{D} \equiv \begin{cases} \text{maximize} & d(x,y) = b^Ty - \frac{1}{2}x^TQx \\ \text{subject to} & Qx + c - A^Ty = s, \quad s \geq 0, \quad y \geq 0 \end{cases} \tag{1.3}$$

where $s \in \Re^n$, $y \in \Re^m$. Note that z and s are primal and dual slack vectors, respectively.

Definition 1.2.1 *A point (x,z) is* feasible *for the problem (1.2) if it satisfies all the linear constraints in (1.2). Analogously a point (x,y,s) is feasible for (1.3) if it satisfies all the linear constraints in (1.3).*
A point (x,z) is said optimal *if in addition it attains the desired minimum.*

The dual problem plays an important role in the design of optimization algorithms, since it provides information about the set of the primal solutions. Furthermore, the dual objective function gives a lower bound on the primal objective function:

Theorem 1.2.1 *(Weak duality) If (x, z) is feasible for the primal and (x, y, s) is feasible for the dual, then*

$$p(x) \geq d(x, y).$$

Proof.

Since $(x, y, s, z) \geq 0$, we have:

$$p(x) = \frac{1}{2}x^TQx+x^Ts-x^TQx+x^TA^Ty = -\frac{1}{2}x^TQx+b^Ty+x^Ts+z^Ty \geq d(x,y).$$

□

Primal and dual objective functions have the same value at an optimal solution.

Theorem 1.2.2 *If the primal problem has an optimal solution (x^*, z^*), then the dual has an optimal solution (x^*, y^*, s^*), such that*

$$p(x^*) = d(x^*, y^*).$$

Since there is no gap in the optimal solution between the primal and the dual objective values, a way to check optimality is to compute the difference between the two objective function values.

Definition 1.2.2 *The* duality gap *of the primal and the dual problems (1.2) and (1.3) is defined as*

$$\Delta = p(x) - d(x, y).$$

Since $b = Ax - z$ implies $b^Ty = (x^TA^T - z^T)y$, we have that:

$$\begin{aligned} &\Delta = p(x) - d(x,y) = \\ &= \frac{1}{2}x^TQx + c^Tx - \left(b^Ty - \frac{1}{2}x^TQx\right) = \\ &= x^Ts + z^Ty \geq 0. \end{aligned} \tag{1.4}$$

Definition 1.2.3 *The primal-dual* feasible set *is*

$$\mathcal{F} = \left\{w = (x, y, s, z) : Ax - b = z, Qx + c - A^Ty = s, w \geq 0\right\}.$$

Definition 1.2.4 *The primal-dual* strictly feasible set *is*

$$\mathcal{F}^0 = \left\{w = (x, y, s, z) : Ax - b = z, Qx + c - A^Ty = s, w > 0\right\}.$$

Note that, since the matrix Q is symmetric positive semidefinite, if the feasible set is a not empty set, then a solution exsists for the QP problem. Furthermore, since the feasible set is a convex set, any local solution of the QP problem is a global minimum.

1.2.2 Optimality conditions

The first-order necessary optimality conditions for the primal and the dual problems, the so-called *Karush-Kuhn-Tucker (KKT) conditions*, are defined in the following theorem [25, 35, 79, 99].

Theorem 1.2.3 *If (x^*, z^*) is a solution of (1.2), there are vectors $y^* \in \Re^m$, $s^* \in \Re^n$, such that the following conditions are satisfied for $(x, y, s, z) = (x^*, y^*, s^*, z^*)$:*

$$\begin{cases} Qx + c - A^T y - s = 0 \\ Ax - z - b = 0 \\ x_i s_i = 0 \quad i = 1, \dots, n \\ y_i z_i = 0 \quad i = 1, \dots, m \\ x \geq 0,\ y \geq 0,\ s \geq 0,\ z \geq 0 \end{cases} .$$

We can restate the nonlinear system of the optimality conditions in the following way:

$$\begin{bmatrix} Qx + c - A^T y - s \\ Ax - z - b \\ XSe \\ YZe \end{bmatrix} = 0, \quad w \geq 0 \tag{1.5}$$

where $w = (x, y, s, z) \in \Re^{(2\times n + 2\times m)}$, X, Y, S, Z are the diagonal matrices of the vectors x, y, s, z, respectively and e is a vector of all ones of appropriate dimension.

The last two equations are the *complementarity conditions*, since they imply that the nonzero elements of the vectors x and s, y and z, must be in complementary locations. Since these conditions have to be satisfied at the solution, the duality gap is equal to zero at the optimal solution.

Note that, since the objective function is convex and the feasible set is convex, the KKT conditions (1.5) are not only necessary, but also sufficient.

1.2.3 The BCQP problem

A special case of convex QP problem is given by the Bound Constrained Quadratic Programming (BCQP) problem, where the only constraints are bounds on the variables:

$$l \leq x \leq u,$$

where l, u are known n-vectors. We assume for simplicity that the bounds are finite.

This class of problems is important in itself since many phisical and engineering problems can be formulated as BCQP, such as obstacle problems, elastic-plastic torsion problems, journal bearing problems (see [73]), but it is also an essential subproblem in many general nonlinear optimization problems.

We transform the BCQP problem to the following form:

$$\begin{cases} \text{minimize} & q(x) = \frac{1}{2}x^TQx + c^Tx \\ \text{subject to} & x + z = e, \quad x \geq 0, \quad z \geq 0 \end{cases} \tag{1.6}$$

where $z \in \Re^n$ and $e \in \Re^n$ is a vector of all ones. The dual of (1.6) can be written as:

$$\begin{cases} \text{maximize} & e^Ty - \frac{1}{2}x^TQx \\ \text{subject to} & Qx + c - y = s, \quad s \geq 0, \quad t = -y \geq 0, \end{cases} \tag{1.7}$$

where $y, s, t \in \Re^n$. The vectors z and s are the *slack* vectors.

Analogously to the case of QP problems, the duality gap of problems (1.6)-(1.7) is

$$\Delta = x^T s + z^T t$$

and the optimality conditions are given by:

$$\begin{bmatrix} Qx + c + t - s \\ x + z - e \\ SXe \\ ZTe \end{bmatrix} = 0, \qquad \begin{bmatrix} x \\ t \\ s \\ z \end{bmatrix} \geq 0.$$

1.3 Interior Point methods for QP problems

1.3.1 Basic idea and main approaches

Interior Point (IP) methods for solving optimization problems have been an active research field since the original work of Karmarkar [53]. They were known since the 1950s [30] and were extensively studied in 1960s [24], but was the publication of the Karmarkar's paper that led to a great interest in IP approach. Indeed, Karmarkar presented a new algorithm for Linear Programming with polynomial complexity, while it was proved that the classical simplex method has an exponential complexity in worst cases. One of the main differences between the new algorithm and the simplex one was that the former moved following a sequence of points belonging to the interior of the feasible set, the latter moved along the boundary of such set, following a sequence of vertices.

Over the years, a growing interest has been devoted to IP methods. Main reasons for the interest in such methods are their computational complexity and good convergence properties and their applicability not only to Linear Programming problems, but also to other classes of optimization problems.

Indeed, even though in the first years work in IP focused mainly on Linear Programming, then IP approach has been extended to several classes of nonlinear programming, semidefinite programming and complementarity problems. For details, see [29, 43, 78, 82, 90, 97, 99, 105] and the references therein.

The basic idea of Interior Point methods is to compute a sequence of approximations of an optimal solution belonging to the interior of the feasible set. Actually, most of the current algorithms generate iterates which stay in the interior of the positive orthant, but are infeasible for linear constraints. A significant advantage of such algorithms over the previous ones is that they do not require an initial feasible point, which is often difficult to compute. These algorithms are called *infeasible* IP algorithms, on the contrary the original ones are called *feasible* IP algorithms. Here we describe only feasible Interior Point algorithms, the basic idea being the same.

Interior Point algorithms can be divided in *primal algorithms*, *dual algorithms*, *primal-dual algorithms.* Primal algorithms are developed purely in terms of primal variables, on the contrary dual algorithms are purely in terms of dual variables. Karmarkar's algorithm and most of algorithms developed in the first years of work made in IP area were primal algorithms. Then researchers focused their attention on primal-dual IP methods, which use explicity both primal and dual variables and became the most successful and powerful class of IP methods.

There are several primal-dual IP algorithms, sharing the same fundamental principles but with different individual features. We briefly describe three primal-dual algorithms: *path-following algorithms*, *potential-reduction algorithms*, *affine-scaling algorithms*, that have been developed for Linear Programming, but can be easily extended to QP and more general problems.

Consider a logarithmic barrier function. A barrier function can be obtained replacing the nonnegativity of constraints in the primal problem by logarithmic barrier penalty terms:

$$L(x,z) = p(x) - \mu \sum_{i=1}^{n} \log x_i - \mu \sum_{j=1}^{m} \log z_j,$$

where $\mu \geq 0$ is a barrier parameter. The first order optimality conditions for the barrier problem

$$\begin{array}{ll} \text{minimize} & L(x,z) \\ \text{subject to} & Ax - b - z = 0 \end{array}$$

can be obtained by considering the associated Lagrangian function

$$\mathcal{L}(x,z,y) = p(x) - \mu \sum_{i=1}^{n} \log x_i - \mu \sum_{j=1}^{m} \log z_j - y^T(Ax - b - z),$$

by differentiating with respect to its variables and by setting $s = \mu X^{-1} e$, $y = \mu Z^{-1} e$:

$$\begin{bmatrix} Qx + c - A^T y - s \\ Ax - z - b \\ XSe \\ YZe \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \mu e \\ \mu e \end{bmatrix} \quad w \geq 0. \tag{1.8}$$

Note that (1.8) is equal to the KKT conditions (1.5) system when $\mu = 0$, so the solution of (1.8) approaches to the solution set of the primal-dual problem as μ goes to zero.
Let $(x_\mu, y_\mu, s_\mu, z_\mu)$ denote the solution of the system (1.8) for a given $\mu > 0$. The trajectory of strictly feasible points:

$$\mathcal{C} = \{(x_\mu, y_\mu, s_\mu, z_\mu) > 0 : \mu > 0\}$$

is called *central path* and is a crucial concept in the theory of primal-dual IP methods.

Path-following algorithms are based on the idea to generate a sequence of points into a neighborhood of the central path, ensuring that the iterates are not too close to the boundary of the feasible set and the positivity condition is not violated, so that the search directions always make a progress toward the solution. At each iteration, a search direction is computed by applying a Newton step to the system (1.8) with a fixed μ, then μ is decreased. The process is repeated until the problem has been solved with the desired accuracy.

The system (1.8) is often written by introducing two parameters $\sigma \in [0, 1]$, $\bar{\mu} = \frac{x^T s + y^T z}{n+m}$, usually named *centering parameter* and *duality measure*, such that $\mu = \sigma\bar{\mu}$. Key ingredients in IP methods are a measure of the quality of a point and a way to "improve" such a point, so a suitable merit function and a way to decrease it are needed. In path-following algorithms the duality measure is used to measure the desirability of a point in the search space: strictly feasible points deviate from the central path $\mathcal{C}$ only because the products $x_i s_i$, $y_i z_i$ are generally not identical, so their deviation is measured by comparing the products with their average value $\bar{\mu}$. Starting from this basic concept, different choices for the neighborhood of the central path are possible. Restrictive choices for the neighborhood allow to obtain only slow progresses at each iteration, leading to short-step path following algorithms; on the contrary, with wide neighborhoods the requirement of staying inside the neighborhood is easier to satisfy and long steps are possible which allow to make faster progress toward the solution at each iteration. A typical choice for the neighborhood of the central path is

$$\mathcal{N}(\gamma) = \left\{ w \in \mathcal{F}^0 : x_i s_i \geq \gamma\bar{\mu},\, y_j z_j \geq \gamma\bar{\mu} \quad \forall i \in \{1, \dots, n\}, j \in \{1, \dots, m\} \right\},$$

with $0 < \gamma < 1$.

A survey on path-following algorithms is given in [43], see also [99].

The framework that we consider for the study of this thesis is a primal-dual Interior Point Potential Reduction method. Here we just outline the

basic idea and some features of such method, while in Chapter 3 we describe in details the Potential Reduction method applied to QP problems.
Potential Reduction algorithms are based on the idea of minimizing a logarithmic function called *potential function*. A single function, the potential function, is used to measure the "optimality" of a point or to decide how to provide a "better" approximation to an optimal solution [90]. Potential Reduction algorithms compute steps of the same form as do path-following algorithms, but they not explicitily follow the central path [99].
The aim to decrease the potential function as much as possible at each iteration of the algorithm can be always achieved under suitable assumptions, and a corresponding bound on the number of iterations necessary to obtain a desired accuracy in the solution can be computed.

Potential Reduction methods are important from an historical point of view because potential functions were used in the development of IP algorithms in the Karmarkar's work and some subsequent works. Main reasons for studying such methods are given by some of their features such as the strong relation between theory and practice, which is often not observed in path-following methods [90]. Potential Reduction methods have good convergence and complexity properties, furthermore they have great flexibility and they are not too hard to implement. Finally, we are interested in Potential Reduction methods since they provide a suitable framework to study the most demanding linear algebra kernels arising in any Interior Point methods.

A variety of potential functions have been considered in developing Potential Reduction algorithms. Primal Potential Reduction methods, based on potential functions that make use of only primal variables, are mainly due to Karmarkar [53] and Gonzaga [42]. Then potential functions taking into account both the primal and the dual variables have been considered. The most successful primal-dual potential function has been introduced by Tanabe [89] and Todd and Ye [91]. First primal-dual Potential Reduction

methods have been developed by Todd and Ye [91] and Ye [104], but the most used primal-dual algorithm is due to Kojima, Mizuno and Yoshise for linear complementarity problems [58]. Primal-dual Potential Reduction methods for convex programming have been applied in [36, 70] and, in the special case of bound constrained quadratic problems, in [48, 17]. Nesterov and Nemirovskii provided some extensions to certain nonlinear problems. Main infeasible versions of the Potential Reduction method for Linear Programmming are due to Mizuno, Kojima and Todd [69] and Tutuncu [92].
A survey on Potential Reduction methods for Linear Programmming and its extensions is given by Todd in [90].

Affine-scaling algorithms, originally introduced by Dikin [18], are historically the simplest IP algorithms. Their basic idea is to transform the original problem in a new one having the variables centered in the feasible set. Given a current iterate x^k, an ellipsoid centered at x^k and inscribed into the feasible set is constructed, then the objective function is minimized on this ellipsoid. In order to transform the problem, the variables are scaled by using diagonal matrices, e.g. x is scaled as $x = D\xi = X^0\xi$, where the scaling matrix maps x to the vector of all 1's e. The denomination of the method is due to this scaling operation.

At a first look, there is no connection with other IP algorithms described above, but a relation can be found by analyzing the directions computed in the search space. The Newton direction δw which is computed by path-following algorithms by applying a Newton step to the system of optimality conditions can be decomposed into two parts [95]:

$$\delta w = \delta w^{opt} + \delta w^{cen}$$

where δw^{opt} is a *step toward optimality*, that tries to improve optimality, and δw^{cen} is a *step toward centrality*, that tries to improve proximity to the central path (in the case of infeasible algorithms there is an additional term that plays the role of a step toward feasibility). The direction computed by an affine-scaling algorithm coincides with δw^{opt}, which indeed is usually called *affine scaling direction*. Hence, affine-scaling algorithms can also be motivated in terms of Newton's method. They consider systems of the form (1.8), where the centering parameter σ is equal to zero, so that the search direction is not forced to go toward the central path.

A general feasible primal-dual IP framework for solving (1.1) is described in Figure 1.1, where $w = (x, y, s, z)$ is an approximation of the solution, $\delta w = (\delta x, \delta y, \delta s, \delta z)$ is a search direction and θ is the step lenght on such a direction.

1.3.2 Linear Algebra in Interior Point methods

Primal-dual IP algorithms compute at each iteration a search direction by applying a Newton step to the nonlinear system (1.8). This system can be viewed as the system of the KKT conditions for the primal-dual problem perturbed with a suitable parameter μ. The linear system deriving by the application of a Newton step has the following form (see Figure 1.1):

$$\begin{bmatrix} Q & -A^T & -I & 0 \\ A & 0 & 0 & -I \\ S & 0 & X & 0 \\ 0 & Z & 0 & Y \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \\ \delta s \\ \delta z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -XSe_n + \mu e_n \\ -YZe_m + \mu e_m \end{bmatrix}. \tag{1.9}$$

We also refer to such system to as the *KKT system*. It is a system of $2n+2m$ equations in $2n+2m$ unknowns, which is nonsingular under the assumption that the matrix A has full row rank. Its unique solution gives a search direction for the considered IP algorithm.

All the IP algorithms described in the previous subsection solve a system

```
! initialization
choose $w^0$ strictly feasible, $\mu > 0$
$k = 0$

! iterations
while (convergence criterion not satisfied) do
    compute a search direction $\delta w^k$ by solving the system:

    $\begin{bmatrix} Q & -A^T & -I & 0 \\ A & 0 & 0 & -I \\ S^k & 0 & X^k & 0 \\ 0 & Z^k & 0 & Y^k \end{bmatrix} \begin{bmatrix} \delta x^k \\ \delta y^k \\ \delta s^k \\ \delta z^k \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -X^k S^k e_n + \mu e_n \\ -Y^k Z^k e_m + \mu e_m \end{bmatrix},$

    compute $\theta^k$ such that $w^k > 0$
    update the approximation of the solution as $w^{k+1} = w^k + \theta^k \delta w^k$
    update $\mu$
    $k = k + 1$
endwhile
```

Figure 1.1: A general feasible primal-dual IP framework.

(1.9) with different choices of the parameter μ. We have already said that in path-following algorithms μ is usually treated as the product of a suitable centering parameter and a duality measure, in potential reduction algorithms μ depends on the current duality gap and on a parameter ρ related to the potential function used by the algorithm, finally in affine scaling algorithms μ is equal to zero.

Note that the first two right hand sides equal to zero denote that the primal and the dual feasibility are kept during the iterations; if an infeasible IP method is used they are equal to the opposite of the dual infeasibility and the primal infeasibility, given by $r_d = Qx + c - A^T y - s$ and $r_p = Ax - b - z$ respectively.

The solution of the system (1.9) is one of the most critical issues in IP methods. A solution could be obtained by factorizing the whole large system and by solving the factorized system, but usually reduced forms to smaller systems are preferred. There are two main stages of reduction that are usually considered and different solution strategies can be developed for the reduced systems. Our aim is to develop efficient strategies to solve the KKT system (1.9) arising at each iteration of a potential reduction algorithm, focusing on iterative tecniques and suitable preconditioning strategies to deal with the ill-conditioning of the system, as we describe in the next chapter.

1.3.3 Interior Point vs. Active Set methods

It is well known that the other main class of numerical methods for Quadratic Optimization is the Active Set one.

The main concept of a method based on an Active Set approach is given by the Active Set at a point x. The *Active set* $\mathcal{A}(x)$ at a point x of a QP problem is defined as the set of the indices of the constraints at which equality holds:

$$\mathcal{A}(x) = \Big\{ i \in \{1, \ldots, m\} : a_i^T x = b_i \Big\}$$

where a_i is the i-th row of the matrix A.
Active Set methods aim to predict which of the inequality constraints are active at an optimal solution, so they are based on the idea of considering at each iteration an estimate of the active set at an optimal solution (see, e.g., [45, 79]). To this aim, at each iteration a set of indices, named *working set*, is obtained by selecting a subset of the constraints that are imposed as equalities, and a quadratic subproblem reduced to the variables whose indices belong to the working set is solved. Hence, equality-constrained QP problems

are solved at each iteration. If the considered estimate of the optimal active set is incorrect, then the current working set is updated and another iteration is performed: if the solution of the solved subproblem violates one of the inequality constraints whose indices are not in the working set, then the working set is updated by adding some constraints, if the solution of the solved subproblem is not that of the original problem, then the working set is updated by deleting some constraints.

Note that the convergence of the Active Set method can be accelerated by combining such method with another optimization method, the *Gradient Projection* method. In [72, 73] this strategy was first proposed for the solution of Bound Constrained Quadratic Programming problems. The standard Active Set strategy is used to explore the “face” of the feasible set defined by the current iterate and the Gradient Projection method is used to move to a different “face”.

From a computational point of view, the IP approach has some advantages over the Active Set one, as observed, e.g., in [45, 85]. Active Set approaches can potentially require a very large number of iterations and, anyway, their number of iterations is often greater than those required for the same problem by Interior Point approaches. Active Set iterations generally do not require a very large computational effort, since the dominant task is given by matrix factorizations, that are usually computed by using suitable strategies to update existing factors and to adapt them to changes in the working set. However, adding or deleting a constraint are operations that make difficult to exploit the structure of the matrices involved in the computations and require the development of ad hoc linear algebra routines. On the other hand, in an Interior-Point structure and dimension of the matrices do not change during the iterative process, so that it is possible to use standard linear algebra routines. Effective advantages of Interior Point implementantions have been shown, e.g., in [7, 71].

1.3.4 Interior Point Software

Over the years, a large number of algorithms and well-estabilished software packages for the solution of optimization problems have been developed. The development of modelling languages, that are useful to easily formulate optimization problems and to use the solvers, and the advent of computational environments for remote solution of optimization problems on the web [23, 76] have provided users with friendly tools to use optimization software. Optimization software is discussed in [74, 101, 7, 45] and on *NEOS Guide to Optimization Software* [77].

As demonstrated by extensive computational comparisons between state-of-the-art software, for nonlinear optimization problems IP-based software is strongly competitive with Active-Set SQP software [7, 71].

We outline currently available IP software packages for QP. Some of them are able to solve also more general nonlinear problems.

Software packages implementing an Interior-Point approach include BPMPD [67, 68], CPLEX/Barrier [51], HOPDM [39], KNITRO [11, 96], LOQO [93, 88], MOSEK [75], OOQP [32], OOPS [40], QPB [15] from the GALAHAD library and Xpress-MP [47].

BPMPD [67, 68] implements an IP approach with advanced presolve techniques. It uses heuristic to make decision between different stages of reduction of the KKT systems and a factorization based on supernodal elimination.

CPLEX/Barrier [51] is based on a primal-dual predictor-corrector method and on Cholesky factorization algorithms for solving KKT systems.

HOPDM [39] implements an infeasible primal-dual IP method with multiple centrality correctors. The solution of the linear systems is based on a fast sparsity-exploiting Cholesky decomposition.

The interesting software KNITRO [11, 96] implements a primal-dual IP method with a trust region approach and uses an iterative solver, namely a projected conjugate gradient, in its computational kernel. Each step is the sum of a normal step to improve the feasibility and a tangential step computed using a projected conjugate gradient iteration. A version known as KNITRO-Direct, based on a direct method to solve the linear algebra kernel, is also available.

LOQO [93, 88] implements an infeasible path-following method, using a factorization of a regularized matrix and a heuristic to choose ordering in reduction of the inner system in order to maximize sparsity in the factorization.

MOSEK [75] is based on a homogeneous monotone complementarity formulation of the problem to be solved and implements an infeasible Interior Point method, using a direct approach to solve the KKT systems. We use MOSEK in order to make a comparison with the performances of two solvers that we have developed to solve convex QP problems, based on an Interior Point Potential Reduction method (see Chapter 4).

OOQP [32] and OOPS [40] are object-oriented IP software. OOQP implements a predictor-corrector algorithm with higher order correction, using a direct approach for solving the linear systems. OOPS is a parallel solver which exploits special structures in the matrices and is able to solve very large problems.

QPB [15] is an IP software that is able to solve convex and non-convex QP problems. It uses a combined conjugate-gradient/Lanczos method with a suitable preconditioner.

Xpress-MP [47] implements a path-following algorithm and is based on Cholesky factorization algorithms for the linear algebra kernels.

Chapter 2

Iterative solution of KKT systems

2.1 Introduction

One of the most critical issues in the development of efficient Interior Point software for large-scale nonlinear optimization problems is the solution of the KKT linear system that arises at each iteration. Therefore, the efficiency and effectiveness of IP algorithms is strongly related to Linear Algebra algorithms which they use. This symbiotic relationship [80] has also motivated recent advances in numerical Linear Algebra.

The aim of this Chapter is to present and to analyze suitable strategies to efficiently solve the linear systems at each iteration of an IP method for convex QP problems. Starting from the perturbed KKT system, we describe its reduced forms, the augmented system and the normal equations, and we discuss about their ill-conditioning. Then we outline some methods to solve such systems. Iterative methods appear to be very promising, mainly when dealing with large-scale problems. We focus on iterative solution of the KKT system in both augmented system and normal equations forms. We use a Conjugate Gradient method with suitable preconditioning strategies, which are different for the two reduced forms. We first consider the normal equations approach and a Conjugate Gradient algorithm with an incomplete Cholesky factorization as preconditioner, then we consider the augmented system approach with a block preconditioner, named Constraint Preconditioner. We present a suitable Constraint Preconditioner for the augmented system matrix arising from convex QP problems with inequality constraints and we analyze the behaviour of this preconditioner with the Conjugate Gradient algorithm. In particular, we show that with a suitable choice of the starting point the Conjugate Gradient algorithm applied to the preconditioned augmented system behaves as if it were applied to suitably preconditioned normal equations. Starting from this equivalence, we prove the convergence results for the Conjugate Gradient method with constraint preconditioner. Finally, we also report a spectral analysis of the preconditioned matrix.

2.2 Approaches to the solution of the KKT system

2.2.1 KKT system reduction

In an IP method to solve QP problems, a Newton direction is obtained at each iteration by solving a $(2n+2m) \times (2n+2m)$ linear system of the form

$$\begin{bmatrix} Q & -A^T & -I & 0 \\ A & 0 & 0 & -I \\ S & 0 & X & 0 \\ 0 & Z & 0 & Y \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \\ \delta s \\ \delta z \end{bmatrix} = \begin{bmatrix} -r_d \\ -r_p \\ -XSe_n + \mu e_n \\ -YZe_m + \mu e_m \end{bmatrix}, \qquad (2.1)$$

where X, S, Y, Z are the diagonal matrices whose diagonal entries are the components of x, s, y, z, respectively, $e_n \in \Re^n$ and $e_m \in \Re^m$ are the vectors of all ones (in the sequel denoted by e, when the dimension is clear from the context), μ is a suitable parameter, r_p, r_d are the primal and the dual infeasibility respectively and are equal to zero in the case of a feasible method (see subsection 1.3.2).

From now on we consider the case of a feasible method since, from a point of view of the solution of the KKT system, the only difference for an infeasible method is in the right hand sides because of r_p, r_d, while the structure of the system is the same.

The solution of the system (2.1) at each IP iteration requires a large computational effort and hence it is one of the most critical issues in an effective implementation of IP methods.

By eliminating δs and δz,

$$\delta z = A\delta x,$$
$$\delta s = Q\delta x - A^T \delta y,$$

the above system is usually reduced to the so-called *augmented system*:

$$\begin{pmatrix} Q+E & -A^T \\ -A & -F \end{pmatrix} \begin{pmatrix} \delta x \\ \delta y \end{pmatrix} = \begin{pmatrix} -Se + X^{-1}\mu e \\ Ze - Y^{-1}\mu e \end{pmatrix}, \tag{2.2}$$

where $E = X^{-1}S$ and $F = Y^{-1}Z$. Since IP methods generate iterates $w = (x, y, s, z) > 0$, E and F have positive diagonal elements.
Note that the linear system (2.2) can also be considered as a system describing a saddle-point problem. Such problems have been extensively studied in recent years, see [8].

A further reduction, obtained by eliminating δy,

$$\delta y = -F^{-1}A\delta x - Ye + Z^{-1}\mu e,$$

leads to the $n \times n$ system of *normal equations (dual ordering form)*:

$$(Q + E + A^T F^{-1} A)\delta x = -Se + X^{-1}\mu e - A^T Ye + A^T Z^{-1}\mu e. \tag{2.3}$$

It is also possible to eliminate δx instead of δy,

$$\delta x = (Q+E)^{-1}A^T\delta y + (Q+E)^{-1}(-Se + X^{-1}\mu e),$$

obtaining the $m \times m$ system of *normal equations (primal ordering form)*:

$$(A(Q+E)^{-1}A^T + F)\delta y = -Ze + Y^{-1}\mu e - A(Q+E)^{-1}(-Se + X^{-1}\mu e). \quad (2.4)$$

The normal equations in primal ordering form have a smaller size then those in dual ordering form, but they require to handle the matrix $(Q+E)^{-1}$, therefore it is more convenient to consider the dual ordering. In the sequel we always consider the normal equations in the dual ordering form.

In [27] it is also shown another equivalent form for the KKT system, the *doubly augmented system*:

$$\begin{pmatrix} Q+E+2A^TF^{-1}A & A^T \\ A & F \end{pmatrix} \begin{pmatrix} \delta x \\ \delta y \end{pmatrix} = \begin{pmatrix} b_1 + 2A^TF^{-1}b_2 \\ b_2 \end{pmatrix},$$

where $b_1 = -Se + X^{-1}\mu e$, $b_2 = -Ze + Y^{-1}\mu e$, and a one-parameter family of linear systems is formulated that includes the augmented system as a special case.

The normal equations have a smaller size than the augmented system and their matrix is symmetric positive definite (spd), while the augmented system one is symmetric indefinite, with n positive and m negative eigenvalues (more precisely, it is quasi-definite [94]). On the other hand, the matrix of the augmented system is sparse if Q and A are sparse, while the one of the normal equations can result dense even if Q and A are sparse, e.g. when A has a dense column, and it is usually denser anyway. Finally, at each IP iteration, the matrix of the normal equations must be entirely recomputed, while only the diagonal coefficients of the augmented system must be updated.

Note that in both augmented system and normal equations, the sparsity pattern of the matrix is unchanged during the whole IP algorithm.
Finally, as the solution at each IP iteration approaches the boundary of the feasible set of the primal-dual problem $(\mathcal{P}, \mathcal{D})$, some entries of E and F can become very large, leading to an increasing ill-conditioning in the normal equations as well as in the augmented system. Ill-conditioning deteriorates the rate of convergence of most iterative methods, like Krylov subspace methods, that have to be used with suitable preconditioners. This is one of the most critical difference with direct methods, as we discuss in the next sections.

2.2.2 Direct vs Iterative solvers

Direct methods are widely used to solve the systems (2.2) and (2.3) in well-established IP software, such as LOQO [93], MOSEK [75], OB1-R [63], PCx [16], OOQP [32]. The Cholesky factorization is usually applied to the normal equations, and variants of the LDL^T factorization, differing essentially for the pivot selection rule, are applied to the augmented system. In both cases, sparse matrix reordering strategies and further "ad hoc" strategies are exploited to deal with the fill-in problem (for a discussion on the direct approach see, e.g., [2, 99]). Note that with direct methods the increasing ill-conditioning of the system matrix is not a severe problem. Indeed, under quite general assumptions, the computed Newton directions are accurate enough to ensure progress toward the optimal solution [26, 98, 100].

When dealing with large-scale problems, the cost of using direct solvers may become prohibitive in terms of both memory and time requirements. In this case, the iterative solvers seem to offer a viable alternative. However, due to the unavoidable ill-conditioning of the linear systems, the use of effective preconditioners is mandatory to obtain useful Newton directions. On the other hand, it is possible to develop ad hoc preconditioners. Moreover iterative solvers allow to use adaptive accuracy requirements in the solution of the system in order to avoid unnecessary iterations when the current IP iteration is far from the solution of the problem.

The first attempts to use iterative solvers are concerned with the application of the preconditioned Conjugate Gradient (CG) method to the normal equations. Diagonal preconditioners or incomplete Cholesky factorizations, either recomputed at each CG step or freezed for a few steps, have been the usual choices [14, 54, 65, 66]. CG with spanning-tree preconditioners [84] and preconditioned LSQR [34] have been also used.

The effectiveness of iterative methods applied to the normal equations can suffer from the loss of sparsity in forming $A^TF^{-1}A$. Therefore, an increasing attention has been devoted to the augmented system, although it is not definite positive. Furthermore, it has been observed in [81] that every preconditioner for the normal equations induces a preconditioner for the augmented system, while the converse is not true. First experiments with iterative methods on the augmented system have been performed using solvers for symmetric indefinite systems, such as SYMMLQ with suitable symmetric preconditioners [33], and a symmetric variant of QMR with a modified SSOR preconditioner [28]. A class of preconditioners reducing the augmented system to positive definite systems, and hence allowing the use of CG, has been proposed in [81].

We consider an iterative solution of the perturbed KKT system, in both augmented system and normal equations forms, based on a preconditioned Conjugate Gradient algorithm. The preconditioner must be obviously chosen so that the preconditioned system is easy to solve and a faster convergence can be achieved. Since the convergence rate of the CG method depends on the distribution of the eigenvalues of the system matrix, the preconditioned matrix should have more favorable spectral properties (properly clustered eigenvalues), than the original one. A detailed description of the preconditioned Conjugate Gradient method is given in [87, 37]. We choose a different form of preconditioning for the CG method applied to the systems (2.2) and (2.3), that we describe in the next sections.

2.2.3 The KKT system for BCQP problems

We note that the KKT system arising in IP methods for BCQP problems has the identity matrix of appropriate dimension in place of the constraint matrix A, so that there are not all the problems related to the presence of such matrix in solving the system, as described in the two previous subsections. The augmented system is:

$$\begin{pmatrix} Q+E & I \\ I & -\bar{F} \end{pmatrix} \begin{pmatrix} \delta x \\ \delta t \end{pmatrix} = \begin{pmatrix} -Se + X^{-1}\mu e \\ Ze - T^{-1}\mu e \end{pmatrix}, \tag{2.5}$$

where $\delta t = -\delta y$, $\bar{F} = T^{-1}Z$ and I is the identity matrix of dimension n, and the normal equations (dual ordering form) are given by:

$$(Q + E + \bar{F}^{-1})\delta x = -Se + X^{-1}\mu e + Te - Z^{-1}\mu e, \tag{2.6}$$

with

$$\delta t = \bar{F}^{-1}\delta x - Te + Z^{-1}\mu e. \tag{2.7}$$

It follows that the normal equations matrix must not be entirely recomputed at each iteration of the IP method, since only E and $\bar{F}$ change at each iteration, affecting only the diagonal elements of the whole coefficient matrix. Furthermore, the normal equations matrix is always sparse as well as the Hessian matrix Q. Hence, the normal equations form is a natural choice for BCQP problems.

2.3 Solution of Normal Equations: CG with an Incomplete Cholesky preconditioner

As noted in the previous section, the normal equations form for the KKT system is very convenient for bound constrained problems. In the first phase of the research activity, our attention was devoted to the solution of BCQP problems with the normal equations approach (see [12]). We consider an iterative solution of the normal equations based on a preconditioned Conjugate Gradient method, where the preconditioner is an incomplete factorization with limited memory introduced in [60].

Incomplete Cholesky factorizations appear as a natural choice for positive definite systems. A general algorithm for an incomplete Cholesky factorization of a matrix B can be obtained by performing the Cholesky factorization and dropping some elements in predetermined positions, so that a decomposition of the following form is computed:

$$B = LL^T - R,$$

where L is a lower triangular matrix and R is the *residual* of the factorization. The matrix $-R$ contains the elements that are dropped during the incomplete elimination process, in order to reduce fill in the Cholesky factor.

The choice of a nonzero pattern for the factor is the fundamental issue in incomplete factorizations. In so-called ILU(p) factorizations, a level of fill is attributed to each element and dropping is based on its value. For an element in location i, j, which is updated by the formula $b_{ij} = b_{ij} - b_{ik} \times b_{kj}$, the level of fill is

$$lev_{ij} = min\left\{lev_{ij}, lev_{ik} + lev_{kj} + 1\right\}$$

and initially $lev_{ij} = 0$ if $b_{ij} \neq 0$, $lev_{ij} = \infty$ otherwise [87]. ILU(p) factorizations are based on the idea of keeping all the elements whose level of fill does not exceed p. The case $p = 0$ gives the ILU(0) factorization, where the nonzero pattern considered for the factorization is the same of the original matrix B. Other strategies to reduce fill-in are based on a *drop tolerance* and drop all elements that are smaller than a prefixed tolerance, so that the elements to drop are chosen on the base of their magnitude instead of their positions.

The above methods have an unpredictable behavior in terms of memory requirements, that is a drawback above all for large-scale problems. Factorizations that avoid unpredictable memory requirements are those of Jones and Plassmann [52], where a certain number of largest elements is retained, and the ILUT factorization [87], depending on a memory parameter and on a drop tolerance. The Incomplete Cholesky Factorization (ICF) with limited and predictable memory requirement by Lin and Moré combines the best features of these factorizations; it was proposed in [60] and successfully applied by the same authors in the TRON software for nonlinear bound-constrained problems [61].

The main feature of the ICF algorithm is that it allows to specify the amount of additional memory available for the incomplete factorization, without the need to use a drop tolerance. Specifically, at the jth step of the process, the jth column of the incomplete factor is determined by retaining the $n_j + p$ largest elements, where p is an user-specified fill-in parameter, i.e. the additional memory of the Cholesky factor, and n_j is the number of nonzero elements in the corresponding column of the original matrix. In contrast with incomplete factorizations that rely on drop tolerance techniques to reduce the fill-in, such strategy allows to obtain an efficient preconditioner with predictable memory behavior, which is desirable from the large-scale optimization point of view.

The incomplete Cholesky factorization may fail for a positive definite matrix, if a negative diagonal element is encountered. A shifting strategy is used in the ICF algorithm in order to avoid negative pivots, so that the matrix $\hat{B} = B + \alpha I$ is factorized, where I is the identity matrix and α is a suitable parameter such that the factorization succeeds. A suitable choice of this parameter allows to handle also indefinite matrices. Note that a diagonal scaling by the l_2-norm of the columns of the initial matrix is applied before computing ICF, but the computed incomplete factor is rescaled back, therefore the preconditioned CG is applied to the original matrix.

The above preconditioning method has been applied to BCQP problems in [12], where the preconditioner P is the incomplete Cholesky factorization of the coefficient matrix.

Numerical experiments show that is useful to perform a scaling of the system in order to reduce the effects of the ill-conditioning. As suggested in [48], a way to tackle such numerical difficulty is to consider the diagonal matrix

$$D = (XZ)^{1/2},$$

in order to scale the equations (2.6)-(2.7) to the form:

$$(D(Q+E)D+G)(D^{-1}\delta x) = D(-Se + X^{-1}\mu e) + D(Te - Z^{-1}\mu e)$$

$$D\delta t = G(D^{-1}\delta x) + D(Te - Z^{-1}\mu e),$$

where $G = D\bar{F}^{-1}D = TX$ is a diagonal positive definite matrix. Thus, we consider for BCQP problems the above scaled systems instead of the original ones.

A crucial issue is the choice of the fill-in parameter p. Since in [61] it has been shown that the use of additional memory often reduces the number of steps drastically, it is useful an adaptive strategy for choosing the value of p to fill the triangular matrix, as we show in Chapter 4.

2.4 Solution of Augmented System: CG with a constrained preconditioner

A very promising class of preconditioners for the augmented system is that of *Constraint Preconditioners (CPs).* They are symmetric indefinite preconditioners having the same block structure as the augmented system matrix, with the upper-right and lower-left blocks unchanged. They date back to [4], but recently they have attracted the interest of many IP researchers and, more generally, of people working on saddle-point problems (see e.g. [9, 19, 22, 38, 49, 55, 59, 62, 83, 86, 27]; an overview of CPs is included in the survey paper [8]). Furthermore, they begin to be implemented in state-of-the-art IP software, such as KNITRO [11, 96] and HOPDM [39].

We consider the following Constraint Preconditioner:

$$P = \begin{pmatrix} diag(Q) + E & -A^T \\ -A & -F \end{pmatrix} \tag{2.8}$$

and its application through sparse direct factorization. Note that, by preconditioning with P the augmented system matrix, all the eigenvalues are moved to the positive semiaxis and at least m of them are made equal to 1 [9, 22].

In the next subsections we prove (see [13]) that, if a suitable CG starting guess is chosen, the approach based on the Conjugate Gradient with Constraint Preconditioner, henceforth referred to as CPCG, is equivalent to applying the CG method with a spd preconditioner to the normal equations in dual ordering form. Starting from this result, we obtain results concerning the convergence of CPCG. Results concerning the behaviour of the preconditioned CG algorithm have been obtained for other preconditioners in the class of constraint preconditioners, see [9, 22, 55, 62, 86] and the recent work [27].

2.4.1 Equivalence between CPCG and a preconditioned CG for Normal Equations

Let us write the augmented system as

$$M\delta u = b, \tag{2.9}$$

where

$$M = \begin{pmatrix} C & -A^T \\ -A & -F \end{pmatrix} \tag{2.10}$$

$C = Q{+}E$, $\delta u = (\delta x^T, \delta y^T)^T$, $b = (b_1^T, b_2^T)^T$, $b_1 = -Se{+}X^{-1}\mu e$ and $b_2 = Ze{-}Y^{-1}\mu e$. From the following factorization of the Constraint Preconditioner,

$$P = \begin{pmatrix} D & -A^T \\ -A & -F \end{pmatrix} = \begin{pmatrix} I & 0 \\ -AD^{-1} & I \end{pmatrix} \begin{pmatrix} D & 0 \\ 0 & -G \end{pmatrix} \begin{pmatrix} I & -D^{-1}A^T \\ 0 & I \end{pmatrix},$$

where $D = diag(Q) + E$ and $G = F + AD^{-1}A^T$ are spd matrices, it results that

$$\begin{aligned} P^{-1} &= \begin{pmatrix} I & D^{-1}A^T \\ 0 & I \end{pmatrix} \begin{pmatrix} D^{-1} & 0 \\ 0 & -G^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ AD^{-1} & I \end{pmatrix} = \\ &= \begin{pmatrix} D^{-1} - D^{-1}A^TG^{-1}AD^{-1} & -D^{-1}A^TG^{-1} \\ -G^{-1}AD^{-1} & -G^{-1} \end{pmatrix}. \end{aligned} \tag{2.11}$$

By exploiting the block structure of the matrix of the augmented system and of P^{-1}, the CPCG algorithm can be written as shown in Figure 2.1, where $\delta u^i = ((\delta x^i)^T, (\delta y^i)^T)^T$ is the approximate solution, $r^i = ((r_1^i)^T, (r_2^i)^T)^T$ the residual, $z^i = ((z_1^i)^T, (z_2^i)^T)^T$ the preconditioned residual and $p^i = ((p_1^i)^T, (p_2^i)^T)^T$ the search direction, at the i-th iterate.

Since the matrices M and P are not spd, a breakdown could occur in this algorithm; furthermore, the CG convergence properties are not guaranteed. However, this is not the case when the starting guess is chosen in such a way that the initial residual has the last m components equal to 0. Indeed, by using such a starting guess, we have that CPCG behaves as if it were applied to a spd system, as stated by the following theorem.

Theorem 2.4.1 *In the CPCG algorithm, if*

$$\delta x^0 = 0, \quad \delta y^0 = -F^{-1} b_2, \tag{2.12}$$

then the vectors $\delta x^i, r_1^i, z_1^i$ and p_1^i are equal to the approximate solution, the residual, the preconditioned residual and the search direction, respectively, at the i-th iteration of the CG algorithm applied to the normal equations

$$(C + A^T F^{-1} A)\delta x = b_1 - A^T F^{-1} b_2, \tag{2.13}$$

with preconditioner

$$\tilde{P} = D + A^T F^{-1} A, \tag{2.14}$$

and null starting guess.

To prove the previous theorem, we use the following lemma (similar results have been provided in [22, 86] for other types of CP):

Lemma 2.4.2 *In the CPCG algorithm, if the starting guess (2.12) is chosen, then, $\forall i \geq 0$,*

$$r_2^i = 0, \tag{2.15}$$

$$t_2^i = 0. \tag{2.16}$$

Proof of Lemma 2.4.2.

The proof proceeds by induction. From (2.12) we have that

$$r_2^0 = b_2 + A\delta x^0 + F\delta y^0 = 0,$$

and hence

$$\begin{aligned} z_2^0 &= -G^{-1}(AD^{-1}r_1^0), \\ z_1^0 &= D^{-1}(r_1^0 + A^T z_2^0) = D^{-1}(I - A^T G^{-1} D^{-1}) r_1^0. \end{aligned}$$

Using these expressions and taking into account that $G = F + AD^{-1}A^T$, we obtain

$$\begin{aligned} t_2^0 &= -Ap_1^0 - Fp_2^0 = -Az_1^0 - Fz_2^0 = \\ &= (-I + AD^{-1}A^T G^{-1} + FG^{-1})AD^{-1}r_1^0 = \\ &= (-I + GG^{-1})AD^{-1}r_1^0 = 0. \end{aligned}$$

Assuming that the relations (2.15) and (2.16) hold for $j \geq 0$, we have

$$r_2^{j+1} = r_2^j - \alpha t_2^j = 0.$$

It follows that

$$\begin{aligned} z_2^{j+1} &= -G^{-1}AD^{-1}r_1^{j+1}, \\ z_1^{j+1} &= D^{-1}(r_1^{j+1} + A^T z_2^{j+1}) = D^{-1}(I - A^T G^{-1} D^{-1}) r_1^{j+1}, && (2.17) \\ p_1^{j+1} &= z_1^{j+1} + \beta p_1^j = D^{-1}(I - A^T G^{-1} D^{-1}) r_1^{j+1} + \beta p_1^j, && (2.18) \\ p_2^{j+1} &= z_2^{j+1} + \beta p_2^j = -G^{-1}AD^{-1}r_1^{j+1} + \beta p_2^j, && (2.19) \end{aligned}$$

and hence

$$\begin{aligned} t_2^{j+1} &= -Ap_1^{j+1} - Fp_2^{j+1} = \\ &= (-I + (AD^{-1}A^T + F)G^{-1})AD^{-1}r_1^{j+1} - \beta(Ap_1^j + Fp_2^j) = \\ &= -\beta t_2^j = 0. \end{aligned}$$

□

We now prove the theorem.

Proof of Theorem 2.4.1.

Let $\delta\tilde{x}^i, \tilde{r}^i, \tilde{z}^i$ and $\tilde{p}^i$ be the approximate solution, the residual, the preconditioned residual and the search direction computed at the i-th iteration of the CG algorithm applied to system (2.13) with preconditioner (2.14). Furthermore, let $\tilde{t}^i = (C + A^T F^{-1} A)\tilde{p}^i$ and let $\tilde{\alpha}^i$ and $\tilde{\beta}^i$ be the scalars used to compute $\delta\tilde{x}^{i+1}, \tilde{r}^{i+1}$ and $\tilde{p}^{i+1}$. We first note that, using the Sherman-Morrison-Woodbury formula, it results

$$\tilde{P}^{-1} = D^{-1} - D^{-1}A^T G^{-1} A D^{-1} \tag{2.20}$$

and hence, if $r_1^i = \tilde{r}^i$, then $z_1^i = \tilde{z}^i$ (see (2.17) in the proof of Lemma 2.4.2). Therefore, we need to prove that, $\forall i \geq 0$,

$$\delta x^i = \delta\tilde{x}^i, \;\; r_1^i = \tilde{r}^i, \;\; p_1^i = \tilde{p}^i. \tag{2.21}$$

To this aim, we prove also that

$$t^i = \tilde{t}^i. \tag{2.22}$$

We proceed by induction. For $i = 0$, the equalities (2.21)-(2.22) come from the choice of the initial guess.
Assuming that the relations (2.21)-(2.22) hold for $i \geq 0$ and recalling Lemma 2.4.2, we readily obtain $\alpha^i = \tilde{\alpha}^i$ and $\beta^i = \tilde{\beta}^i$, and hence

$$\delta x^{i+1} = \delta\tilde{x}^{i+1}, \;\; r_1^{i+1} = \tilde{r}^{i+1}, \;\; p_1^{i+1} = \tilde{p}^{i+1}. \tag{2.23}$$

Furthermore,

$$\begin{aligned} \tilde{t}^{i+1} &= C\tilde{p}^{i+1} + A^T F^{-1} A\tilde{p}^{i+1} = \\ &= C\tilde{z}^{i+1} + A^T F^{-1} A\tilde{z}^{i+1} + \tilde{\beta}^i \tilde{t}^i \end{aligned}$$

and, using (2.18)-(2.19),

$$\begin{aligned} t^{i+1} &= Cp_1^{i+1} - A^T p_2^{i+1} = \\ &= CD^{-1}(I - A^T G^{-1} D^{-1}) r_1^{i+1} + A^T G^{-1} A D^{-1} r_1^{i+1} + \beta^i t^i = \\ &= Cz_1^{i+1} + A^T G^{-1} A D^{-1} r_1^{i+1} + \beta^i t^i. \end{aligned}$$

From (2.23) it follows that

$$A^T G^{-1} A D^{-1} r_1^{i+1} = (I - D\tilde{P}^{-1})\tilde{r}^{i+1} = (\tilde{P} - D)\tilde{z}^{i+1} = A^T F^{-1} A z_1^{i+1};$$

thus

$$t^{i+1} = \tilde{t}^{i+1},$$

which completes the proof. □

2.4.2 Convergence analysis of CPCG

We derive results on the convergence of CPCG from Theorem 2.4.1.

Corollary 2.4.3 *Let $e^i = \delta u^* - \delta u^i$, where $\delta u^* = ((\delta x^*)^T, (\delta y^*)^T)^T$ is the solution of system (2.9), and $\tilde{e}^i = \delta x^* - \delta \tilde{x}^i$. If the starting guess (2.12) is chosen, then*

i) CPCG does not break down;

ii) it converges in at most n iterations;

iii) $(e^i)^T M e^i > 0$ if $e^i \neq 0$.

Then, we can define $||e^i||_M = \sqrt{(e^i)^T M e^i}$ and the following inequality holds:

iv)

$$||e^i||_M \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^i ||e^0||_M,$$

where $\kappa = \lambda_{max}/\lambda_{min}$, with λ_{max} and λ_{min} maximum and minimum eigenvalue of the matrix $\tilde{P}^{-1}(C + A^T F^{-1} A)$.

Proof.

By Theorem 2.4.1, the scalars σ^i and τ^i in CPCG (see Figure 2.1) are equal to the corresponding scalars in the preconditioned CG applied to the normal equations; therefore i) holds. Equality (2.15) in Lemma 2.4.2 yields

$$-A\delta x^i - F\delta y^i = b_2$$

and hence, by using Theorem 2.4.1, if $\delta\tilde{x}^i = \delta x^*$ then $\delta y^i = \delta y^*$, which proves ii). Furthermore, we have

$$(e^i)^T M e^i = (e_1^i)^T C e_1^i + (e_2^i)^T F e_2^i = (\tilde{e}^i)^T (C + A^T F^{-1} A)\tilde{e}^i,$$

which implies iii) and iv). □

The previous corollary shows that M acts on e^i as a spd matrix, although it is indefinite. Note that this does not hold if the matrix F is not positive definite. As already observed in [86], if $F = 0$, e.g. in QP problems with equality constraints, then $||e^i||_M$ is zero whenever $e_1^i = 0$, even if $e_2^i \neq 0$. In this case the CPCG algorithm may not be able to decrease the residual and hence to compute an accurate approximation of the solution. Possible cures to this problem are the diagonal scaling discussed in [86] and the residual update strategy presented in [44]. On the other hand, regularization techniques can be applied to the augmented system to obtain a spd lower right block F [1].

2.4.3 Spectral analysis of the preconditioned matrix

For sake of completeness, we report some results that show the close relationship between the spectrum of $\tilde{P}^{-1}(C + A^T F^{-1} A)$ and that of $P^{-1}M$ and provide a bound on λ_{max} and λ_{min} [22]. By a simple computation we have that

$$P^{-1}M = \begin{pmatrix} I + \bar{G}\bar{D} & 0 \\ -G^{-1}A\bar{D} & I \end{pmatrix},$$

where $\bar{D} = D^{-1}C - I$ and $\bar{G} = I - D^{-1}A^T G^{-1} A$, and that

$$I + \bar{G}\bar{D} = \tilde{P}^{-1}(C + A^T F^{-1} A);$$

therefore m eigenvalues of $P^{-1}M$ are equal to 1, while the remaining n eigenvalues are the ones of $\tilde{P}^{-1}(C + A^T F^{-1} A)$ and hence are real positive. A more complete analysis of the spectrum is given in the following theorem [22].

Theorem 2.4.4 *The matrix $P^{-1}M$ has at least m unit eigenvalues. Furthermore, if the matrix $\bar{G}$ is nonsingular, there are m linearly independent eigenvectors corresponding to these eigenvalues. The remaining eigenvalues of $P^{-1}M$ lie in the interval*

$$\left[\frac{\lambda_{max}(D^{-1}A^TF^{-1}A)+\lambda_{min}(D^{-1}C)}{1+\lambda_{max}(D^{-1}A^TF^{-1}A)}, \frac{\lambda_{min}(D^{-1}A^TF^{-1}A)+\lambda_{max}(D^{-1}C)}{1+\lambda_{min}(D^{-1}A^TF^{-1}A)}\right], \tag{2.24}$$

where $\lambda_{max}(V)$ and $\lambda_{min}(V)$ denote the maximum and the minimum eigenvalue of the matrix V, respectively.

In our case the matrix $\bar{G}$ is nonsingular, since $\bar{G}D^{-1} = D^{-1} - D^{-1}A^TG^{-1}AD^{-1} = \tilde{P}^{-1}$ (see (2.20)). A corollary of the previous theorem shows that the eigenvalues of $P^{-1}M$ are bounded also by the minimum and the maximum eigenvalue of $D^{-1}C$.

Corollary 2.4.5 *The following relation holds:*

$$\lambda_{min}(D^{-1}C) \leq \lambda_{min}(P^{-1}M) \leq \lambda_{max}(P^{-1}M) \leq \lambda_{max}(D^{-1}C).$$

Proof.

The thesis follows from (2.24) by recalling that, if V is a spd matrix, then either $(diag(V))^{-1}V$ has all the eigenvalues equal to 1 or it has at least one eigenvalue less than 1 and one eigenvalue greater than 1 (see, e.g., [86]).[1] □

[1] A direct proof of Corollary 2.4.5 is provided in [9].

```
! initialization
choose $\delta x^0, \delta y^0$;
$i = 0$; $r_1^0 = b_1 - C\delta x^0 + A^T \delta y^0$;   $r_2^0 = b_2 + A\delta x^0 + F\delta y^0$;
$z_2^0 = -G^{-1}(AD^{-1}r_1^0 + r_2^0)$;   $z_1^0 = D^{-1}(r_1^0 + A^T z_2^0)$;
$p_1^0 = z_1^0$;   $p_2^0 = z_2^0$;
$\sigma^0 = (z_1^0)^T r_1^0 + (z_2^0)^T r_2^0$;

! preconditioned CG iterations
**while** ( convergence criterion not satisfied )
   $t_1^i = Cp_1^i - A^T p_2^i$;   $t_2^i = -Ap_1^i - Fp_2^i$;
   $\tau^i = (p_1^i)^T t_1^i + (p_2^i)^T t_2^i$;
   $\alpha^i = \sigma^i / \tau^i$;
   $\delta x^{i+1} = \delta x^i + \alpha^i p_1^i$;   $\delta y^{i+1} = \delta y^i + \alpha^i p_2^i$;
   $r_1^{i+1} = r_1^i - \alpha^i t_1^i$;   $r_2^{i+1} = r_2^i - \alpha^i t_2^i$;
   $z_2^{i+1} = -G^{-1}(AD^{-1}r_1^{i+1} + r_2^{i+1})$;   $z_1^{i+1} = D^{-1}(r_1^{i+1} + A^T z_2^{i+1})$;
   $\sigma^{i+1} = (z_1^{i+1})^T r_1^{i+1} + (z_2^{i+1})^T r_2^{i+1}$;
   $\beta^i = \sigma^{i+1} / \sigma^i$;
   $p_1^{i+1} = z_1^{i+1} + \beta^i p_1^i$;   $p_2^{i+1} = z_2^{i+1} + \beta^i p_2^i$;
   $i = i + 1$;
**endwhile**
```

Figure 2.1: Block formulation of the Conjugate Gradient algorithm with the Constraint Preconditioner.

Chapter 3

Potential Reduction framework

3.1 Introduction

The Interior Point framework that we consider is given by a Potential Reduction method. We already described the motivations for our interest in this method and some its feature in Chapter 1. In this Chapter we describe a primal-dual Potential Reduction (PR) method for solving Quadratic Programming problems. We present the potential function that we consider and its main properties, deriving the basic idea of the PR method to achieve an optimal solution. Starting from the classical convergence results in [58], we report convergence results for QP problems in the case of direct solution of the KKT linear systems arising at each iteration of the method and we extend the convergence analysis to the case of inexact solution of the KKT systems.

3.2 The Potential Reduction algorithm

In this Section we describe the feasible Potential Reduction algorithm that we use for solving QP problems.

In a Potential Reduction algorithm a fundamental role is played by the potential function. We consider the symmetric primal-dual potential function of Tanabe [89] and Todd and Ye [91]:

$$\Phi(w) = \rho \log \left(x^T s + z^T y \right) - \sum_{i=1}^{n} \log \left(x_i s_i \right) - \sum_{j=1}^{m} \log \left(z_j y_j \right), \qquad (3.1)$$

where $w = (x^T, y^T, s^T, z^T)^T$ and $\rho > n + m$ is a suitable parameter.
This logarithmic function has very important properties. In order to examine such properties, it is convenient to rewrite (3.1) in a different way:

$$\begin{aligned} \Phi(w) &= (\rho - (n+m)) \log \left(x^T s + z^T y \right) \\ &\quad - \sum_{i=1}^{n} \log \left(\frac{x_i s_i}{(x^T s + z^T y)/(n+m)} \right) - \sum_{j=1}^{m} \log \left(\frac{z_j y_j}{(x^T s + z^T y)/(n+m)} \right) \\ &\quad + (n+m) \log (n+m) \\ &= \Phi_{opt}(w) + \Phi_{cen}(w), \end{aligned} \qquad (3.2)$$

where

$$\begin{aligned} \Phi_{opt}(w) &= (\rho - (n+m)) \log \left(x^T s + z^T y \right) \\ \Phi_{cen}(w) &= - \sum_{i=1}^{n} \log \left(\frac{x_i s_i}{(x^T s + z^T y)/(n+m)} \right) - \sum_{j=1}^{m} \log \left(\frac{z_j y_j}{(x^T s + z^T y)/(n+m)} \right) \\ &\quad + (n+m) \log (n+m) . \end{aligned}$$

From (3.2) one can notice that the potential function is the sum of two terms related to the basic concept of optimality and centrality, respectively.
The first term Φ_{opt} is related to the optimality, since it tends to $-\infty$ as the iterates tend to optimality, when the duality gap Δ goes to zero.
The second term Φ_{cen} is related to the centrality, that is as better as the pair products $x_i s_i$, $z_i y_i$ are not much smaller than their average value $(x^T s + z^T y)/(n+m)$. Such term of the potential function acts as a barrier when $w = (x^T, y^T, s^T, z^T)^T$ goes toward a point such that $x_i s_i = 0$, $z_i y_i = 0$, but $\Delta > 0$, when the boundary of the feasible set is approached without tending

to an optimal solution.

We can show (see Lemma 3.2.1) that the function Φ_{cen} is bounded from below, so that the potential function approaches $-\infty$ only when Φ_{opt} approaches $-\infty$, that is when Δ tends to zero. Hence, the aim of a Potential Reduction method is to drive the potential function toward $-\infty$ in order to decrease the duality gap to zero. Therefore, the basic idea is to solve the following minimization problem:

$$\begin{array}{ll} \text{minimize} & \Phi(w) \\ \text{subject to} & Ax - b = z, \quad Qx + c - A^T y = s, \quad w > 0. \end{array} \tag{3.3}$$

Lemma 3.2.1 *For any $w > 0$ we have:*

$$\Phi_{cen} \geq (n+m) \log (n+m) \, .$$

Proof.

The function Φ_{cen} can be written as:

$$\Phi_{cen} = -\sum_{i=1}^{n} \log(x_i s_i) - \sum_{j=1}^{m} \log(y_j z_j) + (n+m) \log(x^T s + z^T y).$$

From the arithmetic mean and the geometric mean inequality, we have:

$$\prod_{i=1}^{n} x_i s_i \prod_{j=1}^{m} y_j z_j \leq \left(\frac{1}{n+m} \left(\sum_{i=1}^{n} x_i s_i + \sum_{j=1}^{m} y_j z_j \right) \right)^{n+m}$$

and, taking the logarithm on both sides:

$$\sum_{i=1}^{n} \log(x_i s_i) + \sum_{j=1}^{m} \log(y_j z_j) \quad \leq \quad (n+m) \log \left(\frac{\sum_{i=1}^{n} x_i s_i + \sum_{j=1}^{m} y_j z_j}{n+m} \right),$$

that is

$$(n+m) \log(x^T s + z^T y) - \sum_{i=1}^{n} \log(x_i s_i) - \sum_{j=1}^{m} \log(y_j z_j) \geq (n+m) \log(n+m).\square$$

Lemma 3.2.2 *The potential function is unbounded below on its domain.*

Proof.
For any $\mu > 0$ we can consider a point w_μ on the central path defined by μ. By substituting such point in (3.2) we have:

$$\Phi(w_\mu) = (\rho - (n+m)) \log((n+m)\mu) + (n+m)\log(n+m)$$

that tends to $-\infty$ as μ tends to zero. □

The following Lemma shows the relationship existing between the potential function and the duality gap.

Lemma 3.2.3 *For any strictly feasible point w we have*

$$\Delta \leq \exp\left(\frac{\Phi(w) - (n+m)\log(n+m)}{\rho - (n+m)}\right). \tag{3.4}$$

Proof.
The result follows immediately from (3.2) and Lemma 3.2.1. □

From Lemma 3.2.3, if the PR method generates a sequence $\{w^k\}$ of strictly feasible iterates such that $\Phi(w^k) \to -\infty$, then $\Delta^k \to 0$. Hence, reducing the potential function over the feasible domain forces the sequence to converge to an optimal solution.
Furthermore, if the potential function is reduced by a fixed amount $\delta > 0$ at each iteration of the PR algorithm, that is

$$\Phi\left(w^{k+1}\right) \leq \Phi\left(w^k\right) - \delta, \quad \forall k = 0, 1, 2, \ldots, \tag{3.5}$$

then the duality gap is reduced under a certain threshold ϵ within $O((\rho - (n+m))\log(1/\epsilon))$ iterations [12, 90, 99].

Proposition 3.2.4 *Given a starting point $w^0 \in \mathcal{F}^0$, suppose that the PR algorithm generates a sequence $\left\{w^k\right\} \in \mathcal{F}^0$ that satisfies* (3.5) *for some $\delta > 0$. Then for any $\epsilon \in (0,1)$, we have an index K defined by*

$$K = \left\lceil (1/\delta)(\Phi(w^0) - (n+m)\log(n+m) + (\rho - (n+m))|\log(\epsilon)|) \right\rceil ,$$

such that

$$\Delta^k \leq \epsilon, \quad \forall k \geq K. \tag{3.6}$$

Proof.

By taking the logarithm of both sides in (3.4), we find that (3.6) follows from the inequality:

$$\Phi(w^k) \leq (n+m)\log(n+m) - (\rho - (n+m))|\log(\epsilon)|. \tag{3.7}$$

But, from (3.5), we have

$$\Phi(w^k) \leq \Phi(w^0) - k\delta, \quad k = 1, 2, \ldots$$

so that (3.7) in turn follows from:

$$\Phi(w^0) - k\delta \leq (n+m)\log(n+m) - (\rho - (n+m))|\log(\epsilon)|.$$

The result follows from the last expression. □

We prove that (3.5) holds in Section 3.3.

A constant reduction in the potential function can be achieved, at each iteration, by taking a steepest descent step in the feasible region, with respect to a suitable norm (see, for example, [58, 90]), or, equivalently, by computing a Newton step for the following perturbed KKT equations:

$$H(w^k) = \begin{pmatrix} Qx^k + c - A^T y^k - s^k \\ Ax^k - z^k - b \\ X^k S^k e \\ Y^k Z^k e \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \frac{\Delta^k}{\rho} \\ \frac{\Delta^k}{\rho} \end{pmatrix} \quad w^k > 0, \tag{3.8}$$

where Δ^k is the duality gap corresponding to w^k. This in turn requires the solution of the KKT linear system (2.1), where $r_p = 0$, $r_d = 0$, $\mu = \Delta^k/\rho$:

$$\begin{bmatrix} Q & -A^T & -I & 0 \\ A & 0 & 0 & -I \\ S^k & 0 & X^k & 0 \\ 0 & Z^k & 0 & Y^k \end{bmatrix} \begin{bmatrix} \delta x^k \\ \delta y^k \\ \delta s^k \\ \delta z^k \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -X^k S^k e_n + \frac{\Delta^k}{\rho} e_n \\ -Y^k Z^k e_m + \frac{\Delta^k}{\rho} e_m \end{bmatrix}. \tag{3.9}$$

Once the Newton direction $\delta w^k = (\delta x^k, \delta y^k, \delta s^k, \delta z^k)$ has been computed, the $(k+1)$-th iterate is obtained as:

$$w^{k+1} = w^k + \bar{\theta}^k \delta w^k,$$

where $\bar{\theta}^k$ is a suitable step length. The step length is given by

$$\bar{\theta}^k = \arg \min_{\bar{\theta} \in (0, \theta^k_{max})} \Phi(w^k + \bar{\theta} \delta w^k), \tag{3.10}$$

with

$$\theta^k_{max} = max \left\{ \theta \geq 0 : w^k + \theta \delta w^k \geq 0 \right\}.$$

However, as we point out in Chapter 4, in practice a simpler choice for the step length, that allows to avoid a line search procedure, can be made.

The PR method previously described can be summarized into the algorithmic framework in Figure 3.1.

3.3 Convergence of the Potential Reduction method for QP

In this section we analyze the convergence properties of the feasible Potential Reduction method for Quadratic Programming. We first report the results concerning the "exact" solution of the KKT systems, then we extend these

```
! initialization
choose $w^0$, $\rho > n + m + \sqrt{n+m}$, $\epsilon > 0$
$\Delta^0 = (x^0)^T s^0 + (z^0)^T y^0$
$k = 0$
! PR iterations
while ($\Delta^k \geq \epsilon$) do
  compute $\delta w^k$ by solving the system (3.9)
  compute $\theta^k$ as in (3.10)
  $w^{k+1} = w^k + \theta^k \delta w^k$
  $\Delta^{k+1} = (x^{k+1})^T s^{k+1} + (z^{k+1})^T y^{k+1}$
  $k = k + 1$
endwhile
```

Figure 3.1: The Potential Reduction algorithmic framework.

results to the case of "inexact" solution of the systems, obtained by using iterative methods.

In [58] (see also [99]), it has been proved for Linear Complementarity problems that the reduction in the potential function value can be obtained, under the assumption that the directions are computed exactly. The final result of an elegant convergence theory is the following:

$$\Phi(w^{k+1}) - \Phi(w^k) \leq -(\sqrt{3}/2)\tau + \max\left\{\rho/2n, [2(1-\tau)]^{-1}\right\}\tau^2, \qquad (3.11)$$

where $\rho = n + \sqrt{n}$, $\tau \in (0,1)$. In particular, if $n \geq 2$ and $\tau = 0.4$, then $\Phi(w^{k+1}) - \Phi(w^k) \leq -0.2$.

We note that the optimality conditions for a primal-dual convex QP problem can be stated in terms of a Linear Complementarity problem [57]. Indeed, a Linear Complementarity problem (LCP) is formulated as follows: find an

$(u, v) \in \Re^{2\bar{n}}$ such that

$$v = Mu + q, \quad (u, v) \geq 0 \quad \text{and} \quad u_i v_i = 0,\, i = 1, 2, \ldots, \bar{n},$$

where M is an $\bar{n} \times \bar{n}$ matrix and $q \in \Re^{\bar{n}}$, so that the KKT conditions for the primal-dual pair (1.2)-(1.3) can be interpreted as a LCP problem with dimension $\bar{n} = n + m$, where $u = (x, y)^T$, $v = (s, z)^T$ and M, q are given by

$$M = \begin{pmatrix} Q & -A^T \\ A & 0 \end{pmatrix}, \quad q = \begin{pmatrix} c \\ -b \end{pmatrix}. \tag{3.12}$$

We are particularly interested in solving convex QP problems by using a Potential Reduction framework where the KKT systems are solved with iterative methods, as discussed in Chapter 2. Starting from the convergence results that we have in the case of direct solution of the inner systems, we extend such results to the case of an iterative solution of such systems. Indeed, since iterative methods give approximated solutions of the linear systems, some of the convergence theory has to be reassessed. We show that in both cases, “exact” and “inexact” solution of KKT systems, the relation (3.5) holds, that is the potential function is reduced at least by a constant at each iteration of the Potential Reduction algorithm.

Remark. Let us write the relation (3.4) as

$$\Delta \leq e^{c\,\Phi(w)}\, e^{a} = \bar{a}\, e^{c\,\Phi(w)} \tag{3.13}$$

where $0 < c < 1$ and $a < 0$, so $0 < \bar{a} < 1$.
If at each iteration of the PR algorithm (3.5) holds, then it follows that:

$$\frac{e^{c\,\Phi(w^{k+1})}}{e^{c\,\Phi(w^k)}} = e^{c\left(\Phi(w^{k+1}) - \Phi(w^k)\right)} = e^{-c\,\delta} < 1,$$

hence the sequence $e^{c\,\Phi(w^k)}$ linearly converges to zero.
As a consequence, from (3.13), we have that

$$\Delta^k \to 0 \quad \text{r-linearly.}$$

3.3.1 Preliminaries

We introduce some notation that will be useful in the next subsections. Unless otherwise specified, the symbol $\|\cdot\|$ denotes the Euclidean matrix and vector norms.

$$V_1 = (XS)^{\frac{1}{2}}; \qquad V_2 = (ZY)^{\frac{1}{2}};$$

$$V = diag(V_1, V_2), \qquad V \in \Re^{(n+m)\times(n+m)};$$

$$v_1 = V_1 e_n = [\sqrt{x_i s_i}]_{i=1}^{n}; \quad v_2 = V_2 e_m = [\sqrt{z_i y_i}]_{i=1}^{m}; \quad v = Ve, \ v \in \Re^{n+m};$$

$$h_1 = -v_1 + \frac{\Delta}{\rho} V_1^{-1} e_n; \quad h_2 = -v_2 + \frac{\Delta}{\rho} V_2^{-1} e_m; \quad h = (h_1^T, h_2^T)^T \in \Re^{n+m};$$

$$r = Xr_1 \in \Re^n;$$

$$u_1 = h_1 + V_1^{-1} r; \qquad u_2 = h_2; \qquad u = (u_1^T, u_2^T)^T \in \Re^{n+m};$$

$$D_1 = X^{\frac{1}{2}} S^{-\frac{1}{2}} \qquad D_2 = Z^{\frac{1}{2}} Y^{-\frac{1}{2}}$$

$$v_{min} = \min\{v_1, \ldots, v_{n+m}\} = ||V^{-1}||^{-1}.$$

Note that, from the definition of v, we have:

$$||v||^2 = x^T s + z^T y = \Delta. \tag{3.14}$$

A lower bound exists on $||h||$ for a suitable choice of ρ [58]:

Lemma 3.3.1 *For any $w \in \mathcal{F}^0$ and for $\rho \geq n + m + \sqrt{n+m}$ we have*

$$||h|| \geq \frac{\sqrt{3}}{2v_{min}} \frac{\Delta}{\rho}. \tag{3.15}$$

3.3.2 Exact solution of the KKT systems

We report the convergence results in the case of “exact” solution of the KKT systems in order to introduce the basic ideas of the proof of the convergence and more easily derive the results concerning the iterative solution of such systems.

We restrict our attention to search directions δw which satisfy the equations

$$\begin{cases} X\delta s + S\delta x = -SXe + \frac{\Delta}{\rho}e \\ Y\delta z + Z\delta y = -YZe + \frac{\Delta}{\rho}e \end{cases} \tag{3.16}$$

where, from the first two equations of the system (3.9), we have

$$\delta z = A\delta x, \quad \delta s = Q\delta x - A^T\delta y, \tag{3.17}$$

since we consider a feasible method.

By using the notation introduced in subsection 3.3.1, we can reformulate (3.16) in the following ways:

$$\begin{cases} X\delta s + S\delta x = V_1 h_1 \\ Z\delta y + Y\delta z = V_2 h_2 \end{cases} \tag{3.18}$$

$$\begin{cases} D_1^{-1}\delta x + D_1\delta s = h_1 \\ D_2^{-1}\delta z + D_2\delta y = h_2 \end{cases} \tag{3.19}$$

(by multiplying by V_1^{-1} the first equation and by V_2^{-1} the second equation)

$$\begin{cases} X^{-1}\delta x + S^{-1}\delta s = V_1^{-1}h_1 \\ Z^{-1}\delta z + Y^{-1}\delta y = V_2^{-1}h_2. \end{cases} \tag{3.20}$$

(by multiplying once again by V_1^{-1} the first equation and by V_2^{-1} the second equation).

The following Lemma establishes some useful results.

Lemma 3.3.2 *The following inequalities hold:*

(i) $0 \leq \delta x^T \delta s + \delta z^T \delta y \leq \frac{1}{2}||h||^2$.

(ii) $||X^{-1}\delta x||^2 + ||S^{-1}\delta s||^2 + ||Z^{-1}\delta z||^2 + ||Y^{-1}\delta y||^2 \leq \frac{1}{v_{min}^2}(||h||^2 - 2(\delta x^T \delta s + \delta z^T \delta y))$.

Proof.

We first prove assertion (i). It follows immediately from (3.17) that:

$$\begin{aligned} \delta x^T \delta s + \delta z \delta y &= \delta x^T \delta s + (A\delta x)^T \delta y \\ &= \delta x^T (\delta s + A^T \delta y) \\ &= \delta x^T (Q\delta x - A^T \delta y + A^T \delta y) \\ &= \delta x^T Q \delta x \geq 0 \end{aligned} \tag{3.21}$$

since Q is positive semi-definite.

Furthermore, from (3.19) we obtain that:

$$\begin{aligned} ||h||^2 &= ||h_1||^2 + ||h_2||^2 \\ &= ||D_1^{-1}\delta x + D_1 \delta s||^2 + ||D_2^{-1}\delta z + D_2 \delta y||^2 \\ &= ||D_1^{-1}\delta x||^2 + ||D_1 \delta s||^2 + ||D_2^{-1}\delta z||^2 + ||D_2 \delta y||^2 + 2(\delta x^T \delta s + \delta z^T \delta y) \end{aligned} \tag{3.22}$$

and this implies that

$$||h||^2 \geq 2(\delta x^T \delta s + \delta z^T \delta y). \tag{3.23}$$

Now we prove assertion (ii). From (3.19), (3.20) and (3.22) we obtain:

$$\begin{aligned}
&||X^{-1}\delta x||^2 + ||S^{-1}\delta s||^2 + ||Z^{-1}\delta z||^2 + ||Y^{-1}\delta y||^2 \\
&\quad = ||V_1^{-1}D_1^{-1}\delta x||^2 + ||V_1^{-1}D_1\delta s||^2 + ||V_2^{-1}D_2^{-1}\delta z||^2 + ||V_2^{-1}D_2\delta y||^2 \\
&\quad = ||V_1^{-1}||^2||D_1^{-1}\delta x||^2 + ||V_1^{-1}||^2||D_1\delta s||^2 \\
&\qquad +||V_2^{-1}||^2||D_2^{-1}\delta z||^2 + ||V_2^{-1}||^2||D_2\delta y||^2 \\
&\quad \leq ||V^{-1}||^2(||D_1^{-1}\delta x||^2 + ||D_1\delta s||^2 + ||D_2^{-1}\delta z||^2 + ||D_2\delta y||^2) \\
&\quad = \frac{1}{v_{min}^2}(||h||^2 - 2(\delta x^T\delta s + \delta z^T\delta y)).
\end{aligned} \tag{3.24}$$

□

We impose on the step parameter $\bar{\theta}$ the requirement that

$$\bar{\theta}\max(||X^{-1}\delta x||_\infty, ||S^{-1}\delta s||_\infty, ||Z^{-1}\delta z||_\infty, ||Y^{-1}\delta y||_\infty) = \tau \tag{3.25}$$

for some $\tau \in (0,1)$, which we will determine later.

Next, we evaluate the difference between the potential function values of $\Phi(w^{k+1})$ and $\Phi(w^k)$. By following the same steps described in [58] we obtain that:

$$\Phi(w^{k+1}) - \Phi(w^k) \leq \theta g_1 + \theta^2 g_2, \quad \forall\theta \in (0,\bar{\theta}], \tag{3.26}$$

where

$$g_1 = \frac{\rho}{\Delta}(x^T\delta s + s^T\delta x + z^T\delta y + y^T\delta z) - e_n^T(X^{-1}\delta x + S^{-1}\delta s) - e_m^T(Z^{-1}\delta z + Y^{-1}\delta y) \tag{3.27}$$

$$g_2 = \frac{\rho}{\Delta}(\delta x^T\delta s + \delta z^T\delta y) + \frac{||X^{-1}\delta x||^2 + ||S^{-1}\delta s||^2 + ||Z^{-1}\delta z||^2 + ||Y^{-1}\delta y||^2}{2(1-\tau)}. \tag{3.28}$$

From (3.26), we observe that the quadratic function:

$$f(\theta) = \theta g_1 + \theta^2 g_2,$$

is an upper bound on Φ as a function of the step length θ along the search direction. Therefore, in order to obtain the desired result, we will find bounds on the size of g_1 and g_2.

From the definition of g_2 in (3.28), by using (3.24) of Lemma 3.3.2 we obtain the following bound:

$$g_2 \le \frac{\rho}{\Delta}(\delta x^T \delta s + \delta z^T \delta y) + \frac{1}{2(1-\tau)v_{min}^2}\left(||h||^2 - 2(\delta x^T \delta s + \delta z^T \delta y)\right). \quad (3.29)$$

The function g_1 can be expressed in terms of h. By using (3.18), (3.20), we have:

$$\begin{aligned} g_1 &= \frac{\rho}{\Delta} e_n^T(S\delta x + X\delta s) + \frac{\rho}{\Delta} e_m^T(Z\delta y + Y\delta z) - e_n^T(X^{-1}\delta x + S^{-1}\delta s) \\ &\quad - e_m^T(Z^{-1}\delta z + Y^{-1}\delta y) \\ &= \frac{\rho}{\Delta} e_n^T V_1 h_1 + \frac{\rho}{\Delta} e_m^T V_2 h_2 - e_n^T V_1^{-1} h_1 - e_m^T V_2^{-1} h_2 \\ &= \frac{\rho}{\Delta}\left(V e_{n+m} - \frac{\Delta}{\rho} V^{-1} e_{n+m}\right)^T h \\ &= -\frac{\rho}{\Delta}\left(-v + \frac{\Delta}{\rho} V^{-1} e_{n+m}\right)^T h \\ &= -\frac{\rho}{\Delta}||h||^2. \end{aligned} \quad (3.30)$$

The following theorem holds.

Theorem 3.3.3 *Let*

$$\bar{\theta} = \frac{\tau v_{min}}{||h||} \qquad \text{with} \quad \tau \in (0,1). \quad (3.31)$$

Suppose that the direction δw satisfies the equations (3.16)*. Then, the condition* (3.25) *holds, and*

$$\theta g_1 \le -\frac{\sqrt{3}}{2}\tau \quad (3.32)$$

$$\theta^2 g_2 \leq \max\left\{\frac{\rho}{2(n+m)}, \frac{1}{2(1-\tau)}\right\}\tau^2. \qquad (3.33)$$

Proof.

We show first that the restriction (3.25) holds. From (3.24) we have

$$\begin{aligned} ||X^{-1}\delta x||^2 &\leq \frac{||h||^2}{v_{min}^2} - \frac{2}{v_{min}^2}(\delta x^T\delta s + \delta z^T\delta y) - ||S^{-1}\delta s||^2 - ||Z^{-1}\delta z||^2 - ||Y^{-1}\delta y||^2 \\ &\leq \frac{||h||^2}{v_{min}^2} \end{aligned}$$

so that

$$||X^{-1}\delta x|| \leq \frac{||h||}{v_{min}}$$

and therefore

$$\bar{\theta}||X^{-1}\delta x||_\infty \leq \bar{\theta}||X^{-1}\delta x|| \leq \frac{v_{min}\tau}{||h||}\frac{||h||}{v_{min}} = \tau.$$

The other bounds in (3.25) can be checked in the same way.

We have, from (3.30) and (3.31):

$$\theta g_1 = -\frac{\rho}{\Delta}||h||\tau v_{min}.$$

The inequality (3.32) follows immediately using the bound (3.15).

In order to prove (3.33) we observe that

$$\Delta = ||v||^2 \geq (n+m)v_{min}^2,$$

so that we have from (3.29):

$$\begin{aligned} g_2 &\leq \frac{\rho}{(n+m)v_{min}^2}(\delta x^T\delta s + \delta z^T\delta y) + \frac{||h||^2}{2(1-\tau)v_{min}^2} - \frac{(\delta x^T\delta s + \delta z^T\delta y)}{(1-\tau)v_{min}^2} \\ &= \left\{\left[\frac{\rho}{n+m} - \frac{1}{(1-\tau)}\right]\frac{(\delta x^T\delta s + \delta z^T\delta y)}{||h||^2} + \frac{1}{2(1-\tau)}\right\}\frac{||h||^2}{v_{min}^2} \end{aligned}$$

and therefore

$$\theta^2 g_2 \leq \left\{ \left[\frac{\rho}{n+m} - \frac{1}{(1-\tau)} \right] \frac{(\delta x^T \delta s + \delta z^T \delta y)}{||h||^2} + \frac{1}{2(1-\tau)} \right\} \tau^2.$$

By using assertion (i) of Lemma 3.3.2, if

$$\frac{\rho}{n+m} - \frac{1}{1-\tau} \geq 0$$

then

$$\theta^2 g_2 \leq \left\{ \left(\frac{\rho}{n+m} - \frac{1}{1-\tau} \right) \frac{1}{2} + \frac{1}{2(1-\tau)} \right\} \tau^2 = \frac{\rho}{2(n+m)} \tau^2$$

else

$$\theta^2 g_2 \leq \frac{1}{2(1-\tau)} \tau^2.$$

Thus the inequality (3.33) follows. □

In view of Theorem 3.3.3, from (3.26) we have that:

$$\Phi(w^{k+1}) - \Phi(w^k) \leq -\frac{\sqrt{3}}{2}\tau + \max\left\{ \frac{\rho}{2(n+m)}, \frac{1}{2(1-\tau)} \right\} \tau^2. \qquad (3.34)$$

Note that the reductions obtained in practical implementation are usually much larger than the theoretical lower bound (3.34). This in turn implies that the complexity result, about the lower bound on the number of iterations required to obtain a fixed duality gap reduction, is somewhat pessimistic. It has been verified, by intensive numerical experiments, that the number of iterations of PR algorithm is in practice independent of the problem size (see Chapter 4).

3.3.3 Inexact solution of the KKT systems

Let us consider the KKT system arising at each iteration of the PR algorithm reduced to the augmented system form

$$\begin{cases} (Q + X^{-1}S)\delta x - A^T \delta y = -Se + X^{-1}\dfrac{\Delta}{\rho}e \\ A\delta x + Y^{-1}Z\delta y = -Ze + Y^{-1}\dfrac{\Delta}{\rho}e \end{cases},$$

obtained by eliminating δz, δs from the first two equations of the system (3.9) (see Chapter 2):

$$\delta z = A\delta x, \quad \delta s = Q\delta x - A^T\delta y. \tag{3.35}$$

Suppose that we use an iterative method to solve the system. In Chapter 2 we have shown that, when we use a CPCG method with a suitable choice for the initial solution, the residual corresponding to the second block of equations in the augmented system is equal to zero. Hence, we produce an inexact solution that satisfies the system:

$$\begin{cases} (Q + X^{-1}S)\delta x - A^T \delta y = -Se + X^{-1}\dfrac{\Delta}{\rho}e + r_1 \\ A\delta x + Y^{-1}Z\delta y = -Ze + Y^{-1}\dfrac{\Delta}{\rho}e \end{cases} \tag{3.36}$$

where r_1 is the opposite of the residual vector.
We observe that, by computing δs, δz from (3.35) using the inexact solution of (3.36), we obtain a direction δw that satisfies:

$$\begin{cases} X\delta s + S\delta x = -SXe + \dfrac{\Delta}{\rho}e + Xr_1 \\ Y\delta z + Z\delta y = -YZe + \dfrac{\Delta}{\rho}e \end{cases}, \tag{3.37}$$

i.e. the residual r_1 in the augmented system corresponds to a residual $r = Xr_1$ in the third block of equations of the whole system (3.9). Hence, the approximated computed direction δw is the solution of the system

$$H'(w)\delta w = -H(w) + g + \hat{r}, \tag{3.38}$$

where $H(w)$ is given by (3.8), $g = (0, 0, \frac{\Delta}{\rho}, \frac{\Delta}{\rho})^T$ and $\hat{r} \in \Re^{2n+2m}$ is the opposite of the residual vector on the whole system, which is nonzero, and it is given by $r = Xr_1$, only for the linear equations of the third block of equations, as a consequence of solving inexactly only those equations:

$$\hat{r} = \begin{bmatrix} 0 \\ 0 \\ r \\ 0 \end{bmatrix} \in \Re^{2n+2m}. \tag{3.39}$$

In the remainder of this section our aim is to prove that the convergence theory can be reassessed in order to take into account the inexact solution of the KKT system.

By using the notation introduced in subsection 3.3.1, we can reformulate (3.37) in the following ways:

$$\begin{cases} X\delta s + S\delta x = V_1 u_1 \\ Z\delta y + Y\delta z = V_2 u_2 \end{cases} \tag{3.40}$$

$$\begin{cases} D_1^{-1}\delta x + D_1\delta s = u_1 \\ D_2^{-1}\delta z + D_2\delta y = u_2 \end{cases} \tag{3.41}$$

$$\begin{cases} X^{-1}\delta x + S^{-1}\delta s = V_1^{-1} u_1 \\ Z^{-1}\delta z + Y^{-1}\delta y = V_2^{-1} u_2. \end{cases} \tag{3.42}$$

The following Lemma holds, which is the analogous of Lemma 3.3.2.

Lemma 3.3.4 *The following inequalities hold:*

(i) $0 \leq \delta x^T \delta s + \delta z^T \delta y \leq \frac{1}{2}||u||^2$.

(ii) $||X^{-1}\delta x||^2 + ||S^{-1}\delta s||^2 + ||Z^{-1}\delta z||^2 + ||Y^{-1}\delta y||^2 \leq \frac{1}{v_{min}^2}(||u||^2 - 2(\delta x^T \delta s + \delta z^T \delta y))$.

Proof.

The proof is obtained from that of Lemma 3.3.2 by simply make u play the role of h. □

We aim to find bounds on the size of g_1 and g_2 in (3.26).

From the definition of g_2 in (3.28), by using assertion (ii) of Lemma 3.3.4 we obtain the following bound:

$$g_2 \leq \frac{\rho}{\Delta}(\delta x^T\delta s + \delta z^T\delta y) + \frac{1}{2(1-\tau)v_{min}^2}\Big(||u||^2 - 2(\delta x^T\delta s + \delta z^T\delta y)\Big). \quad (3.43)$$

The function g_1 can be expressed in terms of h and $\bar{r}$, where $\bar{r} = (r^T, 0^T)^T \in \Re^{n+m}$. By using (3.40), (3.42), we have:

$$\begin{aligned} g_1 &= \frac{\rho}{\Delta}e_n^T(S\delta x + X\delta s) + \frac{\rho}{\Delta}e_m^T(Z\delta y + Y\delta z) - e_n^T(X^{-1}\delta x + S^{-1}\delta s) \\ &\quad -e_m^T(Z^{-1}\delta z + Y^{-1}\delta y) \\ &= \frac{\rho}{\Delta}e_n^T V_1 u_1 + \frac{\rho}{\Delta}e_m^T V_2 u_2 - e_n^T V_1^{-1}u_1 - e_m^T V_2^{-1}u_2 \\ &= \frac{\rho}{\Delta}e_{n+m}^T V u - e_{n+m}^T V^{-1}u \\ &= \frac{\rho}{\Delta}\left(Ve_{n+m} - \frac{\Delta}{\rho}V^{-1}e_{n+m}\right)^T u \\ &= -\frac{\rho}{\Delta}\left(-v + \frac{\Delta}{\rho}V^{-1}e_{n+m}\right)^T u \end{aligned}$$

We have:

$$-v + \frac{\Delta}{\rho}V^{-1}e_{n+m} = h$$

and

$$\begin{aligned} h^T u &= h_1^T u_1 + h_2^T u_2 \\ &= h_1^T(h_1 + V_1^{-1}r) + h_2^T h_2 \\ &= ||h_1||^2 + h_1^T V_1^{-1} r + ||h_2||^2 \\ &= ||h||^2 + h_1^T V_1^{-1} r \end{aligned}$$

so that:

$$\begin{aligned} g_1 &= -\frac{\rho}{\Delta}(h^T u) \\ &= -\frac{\rho}{\Delta}(||h||^2 + h^T V^{-1}\bar{r}). \end{aligned} \tag{3.44}$$

Now we have all the tools to find a value of $\bar{\theta}$ and a restriction on $||r||$ that allow us to achieve a constant reduction in Φ.

Theorem 3.3.5 *Let*

$$\bar{\theta} = \frac{\tau v_{min}}{||u||} \qquad \text{with} \quad \tau \in (0,1).$$

Suppose that the direction δw satisfies the equations (3.38). Moreover, suppose that the residual r satisfies

$$||r|| \leq \gamma\frac{\Delta}{\rho}; \quad \gamma < \frac{\sqrt{3}}{2}. \tag{3.45}$$

Then, the condition (3.25) *holds, and*

$$\theta g_1 \leq -\frac{\sqrt{3}-2\gamma}{4}\tau \tag{3.46}$$

$$\theta^2 g_2 \leq \max\left\{\frac{\rho}{2(n+m)}, \frac{1}{2(1-\tau)}\right\}\tau^2. \tag{3.47}$$

Proof.

We show first that the restriction (3.25) holds. From assertion (ii) of Lemma 3.3.4 we have

$$\begin{aligned} ||X^{-1}\delta x||^2 &\leq \frac{||u||^2}{v_{min}^2} - \frac{2}{v_{min}^2}(\delta x^T \delta s + \delta z^T \delta y) - ||S^{-1}\delta s||^2 - ||Z^{-1}\delta z||^2 - ||Y^{-1}\delta y||^2 \\ &\leq \frac{||u||^2}{v_{min}^2} \end{aligned}$$

so that

$$||X^{-1}\delta x|| \leq \frac{||u||}{v_{min}}$$

and therefore

$$\bar{\theta}||X^{-1}\delta x||_\infty \leq \bar{\theta}||X^{-1}\delta x|| \leq \frac{v_{min}\tau}{||u||}\frac{||u||}{v_{min}} = \tau.$$

The other bounds in (3.25) can be checked in the same way.
To show the inequality (3.46), we first observe that from (3.45) and (3.15) it follows that

$$||V^{-1}\bar{r}|| \leq ||V^{-1}|| \cdot ||\bar{r}|| \leq \frac{\gamma}{v_{min}}\frac{\Delta}{\rho} \leq ||h|| \tag{3.48}$$

so that we have:

$$||u|| \leq ||h|| + ||V^{-1}\bar{r}|| < 2||h||, \tag{3.49}$$

$$||h||^2 + h^T V^{-1}\bar{r} \geq ||h||^2 - ||h||||V^{-1}\bar{r}|| = ||h||(||h|| - ||V^{-1}\bar{r}||) > 0. \tag{3.50}$$

The last inequality ensures that the linear coefficient g_1 is negative. Since g_2 is positive, the quadratic approximation $f(\theta)$ is an upturned parabola that takes a dip as θ increases from zero. Furthermore, by using the bounds (3.15), (3.45), (3.49), and (3.50), we have from (3.44):

$$\begin{aligned} \theta g_1 &\leq -\frac{\rho}{\Delta}\frac{||h||}{||u||}(||h|| - ||V^{-1}\bar{r}||)v_{min}\tau \\ &\leq -(\frac{\sqrt{3}}{2} - \gamma)\frac{\tau}{2} = -\frac{\sqrt{3}-2\gamma}{4}\tau. \end{aligned}$$

In order to prove (3.47) we observe that, from (3.29):

$$\begin{aligned} g_2 &\leq \frac{\rho}{(n+m)v_{min}^2}(\delta x^T\delta s + \delta z^T\delta y) + \frac{||u||^2}{2(1-\tau)v_{min}^2} - \frac{(\delta x^T\delta s + \delta z^T\delta y)}{(1-\tau)v_{min}^2} \\ &= \left\{\left[\frac{\rho}{n+m} - \frac{1}{(1-\tau)}\right]\frac{(\delta x^T\delta s + \delta z^T\delta y)}{||u||^2} + \frac{1}{2(1-\tau)}\right\}\frac{||u||^2}{v_{min}^2} \end{aligned}$$

and therefore

$$\theta^2 g_2 \leq \left\{\left[\frac{\rho}{n+m} - \frac{1}{(1-\tau)}\right]\frac{(\delta x^T\delta s + \delta z^T\delta y)}{||u||^2} + \frac{1}{2(1-\tau)}\right\}\tau^2.$$

By using assertion (i) of Lemma 3.3.4, if

$$\frac{\rho}{n+m} - \frac{1}{1-\tau} \geq 0$$

then

$$\theta^2 g_2 \leq \left\{ \left(\frac{\rho}{n+m} - \frac{1}{1-\tau} \right) \frac{1}{2} + \frac{1}{2(1-\tau)} \right\} \tau^2 = \frac{\rho}{2(n+m)} \tau^2$$

else

$$\theta^2 g_2 \leq \frac{1}{2(1-\tau)} \tau^2.$$

Thus the inequality (3.47) follows. □

In view of Theorem above, from (3.26) we have that:

$$\Phi(w^{k+1}) - \Phi(w^k) \leq -\frac{\sqrt{3}-2\gamma}{4}\tau + \max\left\{ \frac{\rho}{2(n+m)}, \frac{1}{2(1-\tau)} \right\} \tau^2.$$

Therefore we have obtained a result that is very similar to that we have in the case of exact solution of the inner systems.
We remark that the bound on the residual, as prescribed by Theorem 3.3.5, can be exploited in the development of an implementation of the PR algorithm in order to adapt the accuracy requirement in the solution of the inner systems to the current value of the duality gap. As the iterates approach the solution, such value can be reduced, so that the accuracy required for the inexact solution increases. Such adaptive strategy is aimed to reduce as much as possible the number of inner iterations, without affecting the number of outer iterations.

Note that, if we use an iterative method to solve the inner system reduced to the normal equations form (dual ordering), then we produce an inexact solution satisfying the system:

$$(Q + X^{-1}S + A^T Z^{-1} Y A)\delta x = -Se + X^{-1}\frac{\Delta}{\rho}e - A^T Y e + A^T Z^{-1}\frac{\Delta}{\rho}e + \tilde{r},$$

where $\tilde{r}$ is the opposite of the residual vector. The residual $\tilde{r}$ corresponds once again to a residual $r = X\tilde{r}$ in the third block of equations of the original system, so that we have the same convergence results that we have proved for an inexact solution of the system in the augmented system form.

Note that, when we consider BCQP problems we can repeat the same analysis of this subsection (see [12]) by considering the following matrices:

$$V = diag(V_1, V_2),\ V \in \Re^{2n\times 2n};\quad V_1 = (XS)^{\frac{1}{2}};\quad V_2 = (ZT)^{\frac{1}{2}};$$
$$D_1 = X^{\frac{1}{2}}S^{-\frac{1}{2}};\qquad D_2 = Z^{\frac{1}{2}}T^{-\frac{1}{2}}.$$

In Theorem 3.3.5 the quantities θg_1, $\theta^2 g_2$ have the following bounds:

$$\theta g_1 \leq -\frac{\sqrt{3}-2\gamma}{4}\tau$$

$$\theta^2 g_2 \leq \max\left\{\frac{\rho}{4n}, \frac{1}{2(1-\tau)}\right\}\tau^2.$$

Chapter 4

Developed software and Numerical results

4.1 Introduction

The Potential Reduction method with iterative solution of the KKT systems has been implemented in two software packages. The first one, named PRBCQP-S, implements a normal equations approach with the incomplete Cholesky factorization described in Section 2.3 for BCQP problems, the second one, named PRQP, implements an augmented system approach with the constraint preconditioner described in Section 2.4 for QP problems of type (1.1). These software packages will be soon available for on line solution of problems on the computational environment for optimization named ESOPO [23], which has been developed in the context of the Italian MIUR FIRB project “Large Scale Nonlinear Optimization”.

In order to describe the developed software packages, we analyze some implementation issues arising in developing efficient and effective IP based software. Our attention is particularly devoted to implementation issues related to the solution of KKT systems with iterative methods. One of the main issues arising in the use of iterative methods for linear systems is the choice of a suitable stopping criterion. We present a computational study of stopping criteria of the preconditioned CG method for solving the KKT systems. We first consider the "simplest" CG termination rule, based on the requirement that the residual norm is less than some factor of the norm of the initial residual, then we consider other criteria, taking into account the convergence results of the Potential Reduction method and the step length on the direction which is obtained by solving the KKT system.

The choice of the fill-in parameter is another relevant implementation issue when dealing with a preconditioner given by an incomplete Cholesky factorization with limited memory.

For more details about implementation issues in IP methods the reader is referred to, e.g., [2, 41].

We show the results of numerical experiments carried out to analyze the effectiveness of the implemented approaches on a set of large-scale quadratic problems. In the context of the PR solver for BCQP problems, we compare the selected ICF preconditioner with other available preconditioners for positive definite systems. In the context of the PR solver for QP problems based on the augmented system approach, the constraint preconditioner is the more promising preconditioner, so we compare the selected iterative approach with a direct approach for the KKT systems. In both BCQP and QP cases, we also compare the developed software packages to the well-estabilished software MOSEK. MOSEK (see also Chapter 1) is designed to solve large-scale optimization problems. Problems that MOSEK can solve include linear problems and quadratic and quadratically constrained problems (integer constrained variables are allowed). For continuous problems MOSEK implements Interior Point based algorithms. Specifically Mehrotra type predictor-corrector methods are considered and the Newton systems arising at each iteration are solved by using a sparse direct method [3].

4.2 PRBCQP-S: a Potential Reduction software for sparse BCQP problems

In this Section we present a software package that solves quadratic problems with only bound constraints. It is based on the Potential Reduction algorithm in Figure 3.1 and solves the KKT systems in normal equations form by applying a preconditioned Conjugate Gradient method where the preconditioner is the incomplete Cholesky factorization with predictable memory requirement by Lin and Morè (Section 2.3).

We analyze the main computational issues, focusing on those associated with the considered iterative approach for the inner iterations, then we present the results of numerical experiments carried out in order to verify the effectiveness of this approach and to make software comparisons to other preconditioners and to the software MOSEK.

4.2.1 Stopping criteria for the inner iterations

The solution of the KKT linear systems in the context of Interior Point approaches is at the inner level of an iterative process which computes, at the outer level, an approximation of the solution of the optimization problem at each iteration. When the approximation of the solution of the optimization problem is far from an optimal solution, at least at the early outer iterations, it is convenient to accept an approximated inner solution having not a high accuracy, that should be required at the last outer iterations. Hence, we devised adaptive stopping criteria, based on the "quality" of the outer iterates.

We analyze different stopping rules for the preconditioned CG algorithm that we use to solve the inner systems. The "simplest" CG stopping criterion is based on the requirement that the residual norm is less than some factor of the initial residual norm:

$$\frac{||r^i||}{||r^0||} \leq tol_{cg}^k,$$

where r^i is the i-th CG step residual, r^0 is the initial residual and tol_{cg}^k is a tolerance at k-th PR iteration.

We adapt the tolerance tol_{cg} to the "quality" of the outer iterates by relating its value to the value of the current duality gap Δ. Since such value decreases when the PR iterates approach the solution, even a higher accuracy is required in computing the Newton direction.
We analyze two adaptive strategies to set the tolerance value:

AD1 $$tol_{cg}^k = \max\left\{\min\left\{\gamma\Delta, tol_{max}\right\}, tol_{min}\right\},$$

AD2 $$tol_{cg}^k = \Delta/\bar{\rho}, \quad \bar{\rho} = n\sqrt{n},$$

where $0 < \gamma < 1$ is a suitable parameter and tol_{min}, tol_{max} are lower and upper bounds on the tolerance. On the base of numerical experiments and taking into account the accuracy required in the stopping criterion for the outer iterations (4.4), we set their values as follows:

$$\begin{array}{rcl} \gamma & = & 10^{-6} \\ tol_{min} & = & 10^{-8} \\ tol_{max} & = & 10^{-4} \end{array} \tag{4.1}$$

The above strategies are both related, by means of a suitable scaling factor, to the value of duality gap Δ at current iterate. Specifically, as scaling factor for Δ the AD2 strategy uses $1/\bar{\rho}$, which is related to the problem size n. As a result, when the problem size increases, the scaling factor decreases, and therefore an higher accuracy is required. In contrast, in the AD1 strategy, the scaling factor is set to γ, independently of n. Furthermore, we force tol_{cg}

to stay in a fixed interval, in order to avoid a too high or too low accuracy. Finally, in our numerical experiments we also consider a termination strategy in which the tolerance tol_{cg} has the fixed value 10^{-9}; we refer to it as the FX strategy.

Numerical results concerning the above stopping criteria are reported in subsection 4.2.8.

4.2.2 Choice of the fill-in parameter

The adaptive strategy for the accuracy requirements in the CG steps is used in conjunction with an adaptive fill-in requirement in the ICF factorization with limited memory described in Section 2.3. Since in [61] it has been shown that the use of additional memory often reduces the number of steps drastically, we consider an adaptive strategy for the choice of the value of the fill-in parameter p. We have selected the following criterion, based on intensive numerical experiments. Starting from the initial value $p = 2$, such value is incremented by 3 if the following condition is satisfied at the k-th outer iteration:

$$step_{cg}^{k}/step_{cg}^{k-1} > 1.2 \quad \text{and} \quad step_{cg}^{k} > 2 \cdot 10^{-3} n,$$

where $step_{cg}^{k}$ is the number of CG steps at the outer iteration k. In few words, we increase the fill-in value if the number of CG steps at a given outer iteration increases at least of more than 20% with respect to the previous iteration.

4.2.3 Step length computation

At each iteration of the PR algorithm, once the Newton direction δw^k has been computed, a steplength θ^k on such direction should be computed by taking (see Section 3.2):

$$\theta^k = \arg \min_{\theta \in (0, \theta^k_{max})} \Phi(w^k + \theta \delta w^k), \tag{4.2}$$

where

$$\theta^k_{max} = max \left\{ \theta \geq 0 : w^k + \theta \delta w^k \geq 0 \right\}.$$

Theoretically a line search procedure is necessary. From a computational point of view, a procedure to compute the step length avoiding the line search should be preferred, mostly when dealing with large-scale problems. We make the following simple choice for the step length:

$$\theta^k = \beta \theta^k_{max}, \tag{4.3}$$

with $0 < \beta < 1$, that leads to a constant reduction in the potential function, as it has been observed in [12, 48].

4.2.4 Starting point selection

When a bound constrained problem is considered, a strictly feasible starting point $w^0 = (x^0, y^0, s^0, z^0)$ can be easily selected. A natural choice for the value of x^0 is the mid point of the two bounds. Then we choose the other values in the following way:

$$\begin{aligned} t^0 &= \alpha e, \quad \alpha = \max\{1, 3 \parallel Qx^0 + c \parallel - (e^T(Qx^0 + c))/2n\}. \\ s^0 &= Qx^0 + c + t^0 \\ z^0 &= e - x^0. \end{aligned}$$

Note that we set x^0, α and t^0 as in [48], where the authors show that such choice implies that $\Delta^0 \leq 1/\epsilon$, which represents a starting condition for their proof of the theoretical convergence of the algorithm.

4.2.5 Other implementation issues

The PR algorithm is stopped at the k-th iteration if the following condition on the relative duality gap is satisfied:

$$\frac{\Delta^k}{1+|q(x^k)|} \leq tol, \tag{4.4}$$

where *tol* is a tolerance specified by the user. A maximum number, *maxit*, of iterations is also considered.

Intensive numerical experiments in [48] suggest that, from the convergence point of view, a good combination of the values for the parameters ρ and β of the PR algorithm is given by $\rho = 0.1n^{1.5}$ and $\beta = 0.99$. We used such values in our implementation.

The values of the algorithmic parameters were set as specified in Table 4.1.

β	*tol*	*maxit*
0.99	1E–6	50

Table 4.1: Values of the algorithmic parameters used in the numerical experiments with PRBCQP-S.

To solve the inner system we use the ICFS package by Lin and Moré, which is based on the ICF algorithm (*http://www-unix.mcs.anl.gov/~more/icfs/*). To perform the remaining computational kernels, namely the inner products and the matrix-vector products, we use respectively the DDOT function from the basic linear algebra subroutines (BLAS) package [20] and the DSSYAX routine from the ICFS package. The matrices are stored by using the compressed columns sparse format, as required by the ICFS package.

4.2.6 Test problems

The test problems we considered are randomly generated BCQP problems and the obstacle problem [73]. For constructing a random (RND) sparse problem, we used a technique similar to that introduced by Moré and Toraldo in [72] to generate random dense problems. The RND problems depend on several parameters: the number of variables n, the condition number (ncond) of the Hessian matrix Q, the magnitude of degeneracy (ndeg) at the solution x^*, the number of active constraints nact(x^*) at the solution x^*, the density of the Hessian matrix (density), the bounds l and u. The Hessian matrix Q is generated by using the Matlab routine *sprandsym*, which returns a random symmetric positive definite matrix with a given condition number and with approximately density×$n \times n$ nonzeros. The vector c is equal to $Qx^* - r$, where r is a vector related to the magnitude of degeneracy ndeg,

$$r_i = [\nabla q(x^*)]_i = \pm 10^{v_i \mathrm{ndeg}}, \quad \forall i: \quad x_i^* = u_i \quad \text{or} \quad x_i^* = l_i$$

where $v_i \in (0,1)$ is randomly generated. We set

- $n = i \times 10^4$, where $i \in \{1,2,3,4,5,6\}$,
- ncond $= 10^l$, where $l \in \{4,6\}$,
- ndeg = -3,
- nact(x^*) $= n/2$,
- density= $10/n$, i.e. the Hessian matrix has about 10 nonzero entries in each row,
- $l_i = 0$ and $u_i = 1$, $\forall i = 1,\dots,n$.

For the obstacle (OBS) problem, Q is a special structured block tridiagonal matrix, with all blocks having the same size t, where $n = t^2$. In particular, the diagonal blocks are also tridiagonal matrices, and the superdiagonal and subdiagonal blocks are equal to $-I$, where I is the identity matrix of size t. The vector c is given by

$$c = -h^2,$$

where $h = 1/(t+1)$, and the bounds are given by

$$l_i = \left(\sin(9.2\alpha_i) \times \sin(9.3\gamma_i)\right)^3,$$

$$u_i = \left(\sin(9.2\alpha_i) \times \sin(9.3\gamma_i)\right)^2 + 0.02,$$

where

$$\alpha_i = (i - \lfloor (i-1)/t \rfloor \times t) \times h \quad \text{and} \quad \gamma_i = \lceil i/t \rceil \times h.$$

In our computational experiments, the OBS problem size is restricted to $i \times 10^4$, where $i \in \{1, 4, 9\}$.

4.2.7 Computational environment

In the next subsections we present the results of numerical experiments carried out on two machines. The first one is an IBM RS6000 workstation available at Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR-CNR, Naples section, Italy). The system is equipped with a Power2 160 MHz microprocessor, 512 MB ECC RAM, and 128 KB data L1 cache. All implementations were carried out using the AIX XL Fortran compiler.

The other system is a PC available at the Department of Mathematics of the Second University of Naples (Caserta, Italy). The system is equipped with a Pentium III 864 MHz microprocessor, 256 MB RAM, 128 KB data L1 cache, and the Linux operating system. All implementations were carried out using the g77 Fortran compiler.

4.2.8 Numerical results: comparison of stopping criteria

The aim of the numerical experiments shown in this subsection is to verify the effectiveness of the adaptive strategies for the accuracy requirement in solving the inner systems proposed in subsection 4.2.1. We compare the AD1 strategy, where the values of the parameters are those specified in (4.1), to the AD2 and to the FX strategies.

Tables 4.2 and 4.3 show the results obtained by using the IBM RS6000 machine to solve the RND problems, with ncond=10^4 and ncond=10^6, respectively. With RNDi we denote the RND problem of size $n = i \times 10^4$. For each strategy, we show the number of outer iterations of the PR algorithm and the total number of preconditioned CG iterations, required to solve all the generated random problems. Moreover, the times expressed in seconds are shown. In the last column, the relative error on the objective function:

$$ERR = \frac{|q(x^*) - q(x^c)|}{|q(x^*)|},$$

where x^* is the solution and x^c is the final computed solution, is shown.

As expected from an Interior Point algorithm, both tables show that the number of outer iterations is almost independent of the problem size, and it is the same for all the considered strategies, showing that the use of an adaptive termination strategy does not affect the number of outer iterations. In Table 4.3 we can observe a general significant rise in the number of CG iterations with respect to Table 4.2 due to the higher ill-conditioning of the Hessian matrix Q, and therefore of the inner system matrices. Moreover, we observe that at least one of the two adaptive strategies gives a number of CG steps less than those required by the FX approach. We also note that, for some problem size, performing more CG steps does not result in higher execution time, since the cost of a step also depends on the fill-in value used in a given outer iteration. However, we can see that an adaptive accuracy requirement leads in general to better performances with respect to the use of a fixed accuracy requirement. In fact, an adaptive strategy allows to require a lower accuracy in the solution of the system in the first PR iterations. We finally note that there is not a clear winner among the two adaptive strategies. It can be only argued that for increasing problem size and ill-conditioning the AD1 strategy should ensure more stable performance than the AD2 strategy.

Problem	*AD1*			*AD2*			*FX*			
	it	*CG it*	*time*	*it*	*CG it*	*time*	*it*	*CG it*	*time*	*ERR*
RND1	18	78	7.60E+0	18	78	7.70E+0	18	105	9.10E+0	8.0E–8
RND2	20	96	2.36E+1	20	118	3.74E+1	20	222	5.90E+1	1.0E–7
RND3	20	129	5.90E+1	20	145	6.11E+1	20	169	4.73E+1	1.0E–7
RND4	22	179	1.06E+2	22	152	9.78E+1	22	214	9.63E+1	1.0E–7
RND5	21	228	1.95E+2	21	229	1.74E+2	21	380	2.53E+2	4.0E–7
RND6	22	442	4.98E+2	22	509	5.31E+2	22	478	6.82E+2	1.0E–7

Table 4.2: Numerical results for RND problems with ncond= 10^4.

4.2.9 Numerical results: comparison with other preconditioners

In Table 4.4 we compare the performance of the PR algorithm based on the ICF preconditioner (PR-ICF) with the performance of the algorithm when the preconditioners SSOR (PR-SSOR) and ILU(0) (PR-ILU), by the ESSL library, are used. We consider the case $ncond = 10^6$ and the AD1 strategy. For each of the three considered preconditioners, we show the number of outer iterations of the PR algorithm, the total number of preconditioned

Problem	*AD1*			*AD2*			*FX*			
	it	*CG it*	*time*	*it*	*CG it*	*time*	*it*	*CG it*	*time*	*ERR*
RND1	19	118	1.88E+1	19	118	1.87E+1	19	179	2.01E+1	1.0E–7
RND2	21	145	6.11E+1	21	162	6.27E+1	21	216	6.30E+1	1.0E–7
RND3	22	187	8.54E+1	22	179	6.87E+1	22	223	6.43E+1	9.0E–8
RND4	23	309	1.95E+2	23	152	9.72E+1	23	341	2.12E+2	3.0E–7
RND5	24	258	2.07E+2	24	465	2.75E+2	24	373	2.48E+2	2.0E–7
RND6	24	491	5.49E+2	24	635	6.12E+2	24	620	6.93E+2	2.0E–7

Table 4.3: Numerical results for RND problems with ncond= 10^6.

CG iterations and the times, expressed in seconds. In the last column, the relative error of the objective function value is reported. These results have been obtained by using the IBM RS6000 machine.

From the results on Table 4.4, we observe that the use of the ILU preconditioner with no additional memory dramatically increases the number of CG steps, and then the times to the solution. This confirms that the use of an adaptive fill-in requirement strategy improves the quality of the incomplete Cholesky factor, and then it drastically improves its performance. When we use the SSOR preconditioner, the number of CG steps is higher than those obtained by using the ICF and ILU(0) preconditioners, but the times are lower than those related to the ILU(0). This is due to the fact that the computation and the use of the SSOR preconditioner, which only requires forward and backward system solutions involving the lower and upper triangular parts of the coefficient matrix, is usually much less expensive with respect to incomplete factorization-based preconditioners, but it leads to poorer performances. Thus, the ICF preconditioner with additional adaptive memory ensures better performance than the SSOR preconditioner.

Problem	*PR-ICF*			*PR-SSOR*			*PR-ILU*			
	it	*CG it*	*time*	*it*	*CG it*	*time*	*it*	*CG it*	*time*	*ERR*
RND1	19	118	1.88E+1	19	4140	4.90E+1	19	3299	8.72E+1	1.0E–7
RND2	21	145	6.11E+1	21	5215	1.77E+2	21	4245	3.37E+2	1.0E–7
RND3	22	187	8.54E+1	22	6590	3.62E+2	22	9066	1.05E+3	9.0E–8
RND4	23	309	1.95E+2	23	6199	5.35E+2	23	6063	1.19E+3	3.0E–7
RND5	24	258	2.07E+2	24	8464	9.53E+2	24	6792	1.71E+3	2.0E–7
RND6	24	491	5.49E+2	24	7059	1.16E+3	24	6726	2.52E+3	2.0E–7

Table 4.4: Numerical results for different preconditioners and for RND problems with ncond= 10^6.

4.2.10 Numerical results: comparison with MOSEK

The aim of this subsection is to present numerical results of a comparison between the PRBCQP-S software and the MOSEK [75] software. Due to the current unavailability of such software on the IBM R6000 system, the results refer to experiments carried out on a PC. We consider the version of the PR-ICF solver with the AD1 tolerance strategy. In order to have computed solutions with the same accuracy of those provided by MOSEK, we use the value 10^{-7} into the outer stopping criterion. Tables 4.5 and 4.6 show the results to solve the RND problems, with ncond= 10^4 and ncond= 10^6, respectively. For each of the two solvers, we show the number of outer iterations (and the total number of preconditioned CG iterations for the PR-ICF algorithm) and the overall CPU times expressed in seconds.

We observe an increase in the number of iterations (both outer and inner) with respect to the results shown in Table 4.2 and 4.3. This is due to the higher accuracy demand on the quality of the solution. MOSEK gives better performances than PR-ICF for problems with size $n < 4 \times 10^4$. The scenario changes when we solve higher dimension problems. Such results show that for large sparse problems without any structure, the use of preconditioned iterative methods is competitive with the use of sparse direct ones.

Finally, we show the results obtained by solving the obstacle problems with PR-ICF solver and with MOSEK. Due to the special structure of such problems, we use a modified AD1 strategy in order to require lower accuracy in solving the linear systems. Specifically, we use:

$$\begin{array}{rcl} \gamma & = & 10^{-2} \\ tol_{min} & = & 10^{-5} \\ tol_{max} & = & 10^{-2} \end{array} .$$

Furthermore, we let the fill-in parameter p to have only the value 1 or 2. In Table 4.7 we report the number of outer iterations (and the total number of preconditioned CG iterations for the PR algorithm) and the times expressed in seconds, needed for solving the OBSi problems, where $i \in \{1, 4, 9\}$.

We observe that also for high structured problems, for which a sparse direct inner solver is very stable and efficient, the use of iterative approaches compares favourably.

Problem	*PR-ICF*			*MOSEK*	
	it	*CG it*	*time*	*it*	*time*
RND1	19	83	4.80E+0	14	3.40E+0
RND2	21	114	1.76E+1	14	1.68E+1
RND3	21	136	2.90E+1	15	2.34E+1
RND4	23	187	5.32E+1	15	8.94E+1
RND5	23	272	1.21E+2	16	1.61E+2

Table 4.5: Comparison of PR-ICF and MOSEK for RND problems with ncond= 10^4.

Problem	*PR-ICF*			*MOSEK*	
	it	*CG it*	*time*	*it*	*time*
RND1	21	129	1.16E+1	16	3.80E+0
RND2	22	152	3.32E+1	17	1.99E+1
RND3	24	208	4.65E+1	17	2.59E+1
RND4	25	339	9.48E+1	17	1.02E+2
RND5	26	305	1.18E+2	18	1.80E+2

Table 4.6: Comparison of PR-ICF and MOSEK for RND problems with ncond= 10^6.

Problem	*PR-ICF*			*MOSEK*	
	it	*CG it*	*time*	*it*	*time*
OBS1	14	144	2.70E+0	11	3.60E+0
OBS4	16	319	2.36E+1	13	2.39E+1
OBS9	18	595	8.58E+1	14	8.09E+1

Table 4.7: Comparison of PR-ICF and MOSEK for OBS problems.

4.3 PRQP: a Potential Reduction software for sparse convex QP problems

In this Section we present a software package for solving the quadratic optimization problem (1.1). It is based on the Potential Reduction algorithm in Figure 3.1 and solves the KKT systems in augmented system form by applying a preconditioned Conjugate Gradient method where the preconditioner is the constraint preconditioner introduced in Section 2.4.

As we made for the previous solver, we analyze the computational issues related to the solution of the linear systems with an iterative method. In particular, we make a computational study of stopping criteria for the preconditioned CG algorithm that we use to solve the KKT systems, extending the experiences that we made in the case of BCQP problems. Then we present the results of numerical experiments, showing results concerning different stopping criteria for the CG steps and the results of comparisons to a direct approach for the KKT systems and to the software MOSEK.

4.3.1 Stopping criteria for the inner iterations

In order to make a computational study of stopping criteria of the preconditioned CG algorithm for solving the linear systems, we first consider a criterion that is in analogy to that used in the context of the PR algorithm for BCQP problems (subsection 4.2.1), then we rely the stopping rule of the CG steps to the convergence properties of the PR algorithm.

As we have already noted, the “simplest” criterion to stop CG steps is based on the reduction of the relative residual of the approximated solution:

$$\frac{||r^i||}{||r^0||} \leq tol_{cg}^k, \tag{4.5}$$

where r^i is the i-th CG step residual at k-th PR iteration, r^0 is the initial residual and tol_{cg}^k is a tolerance at k-th PR iteration.

We consider, in analogy to the case of BCQP problems (see subsection 4.2.1), tol_{cg}^k defined as

$$tol_{cg}^k = \max\left\{\min\left\{\gamma\Delta, tol_{max}\right\}, tol_{min}\right\}, \tag{4.6}$$

where $0 < \gamma < 1$ and tol_{min}, tol_{max} are suitable tolerances, so that the accuracy of the solution of the augmented system is related to the current value of the duality gap, through the scaling factor γ; the aim is to avoid unnecessary CG steps far from the solution of the optimization problem, without affecting significantly the number of PR iterations. The bounds tol_{min} and tol_{max} are used to prevent from too large or too low accuracy requirements. The values of γ, tol_{min} and tol_{max} are set as follows:

$$\begin{array}{rcl} \gamma & = & 10^{-6} \\ tol_{min} & = & 10^{-8} \\ tol_{max} & = & 10^{-4} \end{array} . \tag{4.7}$$

Intensive numerical experiments made with the above stopping criterion showed that the performance of the preconditioned CG method strongly depend on the value of the parameter γ and the values of the bound tolerances. Furthermore, a suitable combination of these values, that seems to be the best one, leads to use the upper bound tolerance much more times than the original value $\gamma\Delta$.

We tried to devise another stopping criterion which was not depending on parameters based on computational experiments. Theoretical results suggest a strategy for the choice of a stopping criterion, allowing to rely such criterion to the convergence properties of the considered Interior Point framework. From Theorem 3.3.5 we have that, when we use an iterative method to solve the KKT systems, we have a constant reduction in the potential function if the (absolute) residual norm $||\hat{r}||$ on the whole KKT system satisfies an upper bound depending by the duality gap and by the parameter ρ, which corresponds to the perturbation of the original KKT system. This bound on the residual norm can be used as a stopping criterion for the CG steps.

We note that the same criterion can be derived also starting from another point of view, only taking into account the behaviour of inexact Newton methods. Consider the original KKT equations and the perturbed ones:

$$F(w^k) = \begin{pmatrix} Qx^k + c - A^T y^k - s^k \\ Ax^k - z^k - b \\ X^k S^k e \\ Y^k Z^k e \end{pmatrix} = 0, \quad w^k > 0,$$

$$F_\mu(w^k) = \begin{pmatrix} Qx^k + c - A^T y^k - s^k \\ Ax^k - z^k - b \\ X^k S^k e - \mu e_n \\ Y^k Z^k e - \mu e_m \end{pmatrix} = 0, \quad w^k > 0.$$

In [5] it is observed that an inexact Newton step applied to the system of perturbed KKT equations, that gives

$$F'_\mu(w)\delta w = -F_\mu(w) + \hat{r},$$

where $\hat{r} \in \Re^{2n+2m}$ is the opposite of the residual vector, can be interpreted as an inexact Newton step applied to the original system:

$$F'(w)\delta w = -F(w) + r_g,$$

where

$$r_g = \mu \left(0, 0, e_n^T, e_m^T\right)^T + \hat{r}.$$

It is also shown that, if

$$||\hat{r}|| \leq \eta\Delta,$$

taking into account the observation that [6]

$$||F(w)|| \geq \frac{\Delta}{\sqrt{n+m}},$$

then it can be found a forcing sequence of the inexact Newton method applied to the original KKT equations system. Indeed, we have

$$||r_g|| \leq \mu\sqrt{n+m} + ||\hat{r}||$$

and, since in a PR algorithm $\mu = \Delta/\rho$, we find

$$||r_g|| \leq \left(\frac{n+m}{\rho} + \eta\sqrt{n+m}\right)||F(w)||,$$

where $(n+m)/\rho + \eta\sqrt{n+m}$, with η such that $(n+m)/\rho + \eta\sqrt{n+m} < 1$, is a forcing sequence. Hence we have the following bound on $||\hat{r}||$:

$$||\hat{r}|| \leq \eta\Delta, \quad \eta < \frac{1-(n+m)/\rho}{\sqrt{n+m}}$$

that is

$$||\hat{r}|| \leq \frac{\Delta}{\rho},$$

which is the same bound obtained from the convergence theory of the PR method with inexact solution of the inner systems.

We recall that the only nonzero components of the residual $\hat{r}$ are those corresponding to the third block of equations, say $\tilde{r}$, and the residual r in the first block of equations of the KKT system reduced to the augmented system corresponds to a residual $\tilde{r} = Xr$ in the third block of equations of the whole system (see Chapter 3). Then, in accordance with the theoretical results about the convergence of the PR method, the preconditioned CG algorithm could be stopped when

$$||X^k r^i|| \leq \frac{\Delta}{\rho}. \tag{4.8}$$

where r^i is the i-th CG step residual in the first block of equations of the augmented system at k-th PR iteration.

Note that the quantity $||Xr||$ strongly depends on $||X||$. We have experimentally observed that, when the norm of X is very large at the first

PR iterations, due to the choice of the starting point, we have a high accuracy requirement in the solution of the augmented system by using (4.8) as a stopping criterion. This leads to a large number of CG steps, that are computationally expensive and are not useful at the first PR iterations. Therefore, we consider the following stopping criterion for the preconditioned CG steps, where we avoid the dipendence on X, so that we reduce the accuracy requirement at the first outer iterations:

$$||r^i|| \leq tol_{cg}^k, \tag{4.9}$$

where tol_{cg}^k is given by

$$tol_{cg}^k = \frac{\Delta}{\rho}. \tag{4.10}$$

As in the first criterion we considered, tol_{cg}^k is adapted to the outer iterations in such a way that the accuracy requirement grows up as the duality gap decreases.

The results of numerical experiments, that we made by using the preconditioned CG method with the above stopping criterion, show that this method behaves well on the most of the considered problems (see subsection 4.3.5). This motivates our investigation on the use of the preconditioned CG method and a further search of a good stopping criterion in order to improve its performance.

We tried to use of a stopping criterion based on the sufficient condition given by the convergence theory, reducing the accuracy requirement in (4.9). To this aim, the tolerance is scaled by a constant c greater than 1:

$$tol_{cg}^k = c\,\frac{\Delta^k}{\rho}. \tag{4.11}$$

A numerical comparison of the considered stopping criteria is in subsection 4.3.5. We show that the last criterion allows to obtain the best performance, but a suitable safeguard strategy must be used to avoid that inaccurate directions deteriorate the “quality” of the PR iterates.

4.3.2 Other implementation issues

The step length computation is made as described in subsection 4.2.3 for the PRBCQP-S software.

The selection of a good starting point for an Interior Point algorithm for nonlinear optimization is still an open problem. Some heuristics to find a strictly feasible point have been developed that work well in practice, based on the solution of least-squares problems; another starting point strategy is proposed in [31]. For a discussion about the starting point selection in IP methods see, e.g., [2, 99]. This issue is not addressed in this thesis. When starting points are not available, they are computed by solving "hand-tuned" least squares problems.

As for the PR algorithm for BCQP problems, the outer iterations are stopped if the following condition on the relative duality gap is satisfied:

$$\frac{\Delta^k}{1+|q(x^k)|} \leq tol, \tag{4.12}$$

where *tol* is a tolerance specified by the user, and a maximum number, $maxit$, of iterations is also considered.

The parameter ρ in the potential function (3.1) is set as

$$\rho = \gamma_\rho(n+m), \tag{4.13}$$

where $\gamma_\rho \geq 2$ is selected taking into account the size of the problem. This choice is driven by the observation that a value of ρ larger than the lower bound value $n+m+\sqrt{n+m}$ usually produces better numerical performance, although it degrades the theoretical complexity estimate (see, e.g., [99]).

For all the test problems, the algorithmic parameters involved in the definition of the potential function (see (4.13)), of the PR step length (see (4.3)) and of the stopping criterion for the outer iterations (see (4.12)) were set as specified in Table 4.8. The value of β was chosen as suggested in [48]; the other values were selected on the basis of numerical experiments.

γ_ρ		β	*tol*	*maxit*
6	if $m + n \leq 15000$	0.99	1E–7	50
10	otherwise			

Table 4.8: Values of the algorithmic parameters used in the numerical experiments with PRQP.

At each PR iteration the Constraint Preconditioner is factorized and applied using the MA27 suite of routines from the Harwell Subroutine Library, which implements a variant of the Gaussian elimination to compute the LDL^T factorization of a symmetric sparse matrix [21].
The preconditioned CG algorithm is implemented using (2.12) as starting guess. The block structure of P is not exploited as in the algorithm in Figure 2.1, since this requires the explicit computation of G.
Ad hoc routines have been developed to compute sparse symmetric and unsymmetric matrix by vector products.

4.3.3 Test problems

Two sets of test problems were considered. The first one includes 15 QP problems from the CUTEr collection [46], that were modified to have only linear inequality constraints and nonnegative variables with no upper bounds. The second set is composed of 9 problems that were randomly generated. The Hessian matrices were built by using the *sprandsym* Matlab function, requiring about $10n$ nonzero entries, which, for the selected values of n, correspond to more than 99% of sparsity; the constraint matrices were obtained by using the random matrix generator from the *Matrix Market Deli* collection (http://math.nist.gov/MatrixMarket/deli/Random), requiring a sparsity of at least 98%. The dimensions n and m, the number of nonzero entries of the upper (or lower) triangle of the Hessian matrix Q, $nnz(Q)$, and the number of nonzero entries of the constraint matrix A, $nnz(A)$, of the two set of problems are reported in Table 4.9.

For the modified CUTEr problems, strictly feasible starting vectors x^0 and y^0 were computed by solving "hand-tuned" least squares problems. In the case of the random tests, the vectors b and c in the primal-dual problem (1.2)-(1.3) were defined in such a way that $x^0 = e_n$ and $y^0 = e_m$ resulted strictly feasible.

4.3.4 Computational environment

All the tests were performed using a personal computer available at the Department of Mathematics of the Second University of Naples, equipped with a 2.53 GHz Pentium IV processor, a memory of 1.256 GB and a L2 cache of 512 KB, and running the Linux Red Hat 9.0 operating system. The PR software is coded in Fortran 77, with double precision. It was compiled using the g77 3.2.2 Fortran compiler with the -O3 option; the version 3.1.1.42 of MOSEK was used.

4.3.5 Numerical results: comparison of stopping criteria

We show the results of numerical experiments carried out to analyze the behaviour of the PR implementation based on CPCG (henceforth referred to as PR-CPCG) with the stopping criteria described in subsection 4.3.1. We show only results concerning the two last criteria, based on convergence properties of the PR method. We restate here the considered criteria for semplicity of description:

Problem	n	m	$nnz(Q)$	$nnz(A)$
AUG3DCQP	3873	1000	3873	6546
CVXQP1-a	1000	500	3984	1498
CVXQP1-b	10000	5000	39984	14998
CVXQP2-a	1000	250	3984	749
CVXQP2-b	10000	2500	39984	7499
CVXQP3-a	1000	750	3984	2247
CVXQP3-b	10000	7500	39984	22497
GOULDQP3-a	1999	999	3996	2997
GOULDQP3-b	9999	4999	19996	14997
GOULDQP3-c	19999	9999	39996	29997
MOSARQP1	2500	700	2545	3422
MOSARQP2	2500	700	2545	3422
STCQP2-a	1025	510	5615	2805
STCQP2-b	4097	2052	26603	13338
STCQP2-c	8193	4095	57333	28665
RAND1	20000	100	105001	2000
RAND2	20000	1000	105001	4000
RAND3	30000	1000	143250	3000
RAND4	30000	1000	143250	6000
RAND5	30000	3000	143250	9000
RAND6	30000	5000	143250	15000
RAND7	30000	8000	143250	24000
RAND8	40000	1000	210048	8000
RAND9	50000	1000	262525	10000

Table 4.9: Dimensions and number of nonzero entries of the modified CUTEr test problems (AUG3DCQP - STCQP2-c) and of the random test problems (RAND1 - RAND9).

STOP1 $$||r^i|| \leq tol_{cg}^k, \quad tol_{cg}^k = \frac{\Delta}{\rho}$$

STOP2 $$||r^i|| \leq tol_{cg}^k, \quad tol_{cg}^k = c\,\frac{\Delta}{\rho}, \;\; c = 10.$$

According to the results presented in Section 2.4, n is always set as maximum number of CG steps.

In Table 4.10 we report the number of PR iterations and the total number of CG iterations performed by PR-CPCG when the STOP1 and the STOP2 criteria are used. Results concerning the STOP1 criterion show that, as expected from an Interior Point algorithm, the number of PR iterations is almost independent of the problem size. The problem AUG3DCQP has a diagonal Hessian, so it requires a very small number of CG iterations. On the contrary, the GOULDQP3 problems require a very large number of CG iterations. A more detailed analysis of the behaviour of PR-CPCG shows that at least 94% of the CG iterations are carried out in the last three PR iterations. A possible explanation is that these problems have no active constraints at the computed solution; therefore, the entries of $F = Y^{-1}Z$ eventually become all very large, leading to very ill-conditioned augumented systems. However, the behaviour of PR-CPCG in this case deserves further investigation.

Results concerning the STOP2 criterion show that, when we reduce the accuracy requirement on the CG solutions, the number of PR iterations dramatically increases for some of the test problems, and for 7 problems the maximum number of allowed outer iterations is reached without achieving an optimal solution.

In order to better analyze the reasons of these failures, we consider, at each outer iteration of PR-CPCG with STOP2 criterion, not only the step length θ on the direction computed with CPCG, but also the step length θ' on the direction computed with a direct method starting from the same point used in CPCG, and we compute the angle α between the two directions. We show these details for two selected test problems in Table 4.11 and Tables 4.12-4.13. For the first one the convergence is reached in a significantly larger number of iterations with respect to the STOP1 case, for the second one the convergence is not obtained. We report for each PR iterate the number of CG steps, the value of the primal objective function, the step lengths θ and $\theta^{'}$, the angle α between the directions corresponding to the two step lengths.

We note that the step length θ becomes shorter and shorter, so that the quality of the corresponding iterates degrades, making difficult to reduce the value of the objective function and to achieve the convergence to an optimal solution. In the sequel we describe the possible reasons for this drawback and a strategy that we developed as a possible cure.

A strategy based on restarting CG iterations

A possible reason for the above problem is that high values of tol_{cg} lead to poorly centered iterates. A point $w = (x, y, s, z)$ is a *centered* point if all the pair products $x_i s_i$, $y_i z_i$ are equal to the centering measure (see Chapter 1) $\Delta/(n+m)$. Hence, when

$$v_{min} = min\,(x_i s_i, y_j z_j) << \frac{\Delta}{n+m}, \quad i \in \{1, \ldots, n\}\,, j \in \{1, \ldots, m\} \quad (4.14)$$

we have a poorly centered point w.
In correspondence to a point w such that (4.14) holds, the step length tends to be short, since from (3.31) we have:

$$\theta \leq 2\tau \frac{\rho}{\sqrt{3}} \frac{v_{min}}{\Delta} << 2\tau \frac{\rho}{\sqrt{3}(n+m)}, \quad \tau \in (0, 1).$$

In order to avoid inaccurate directions, that deteriorate the “quality” of the PR iterates, we developed a strategy based on restarting CG iterations:

at each PR iteration, if the computed step length θ is "too short", then it is rejected and we restart CG from the last computed direction by using an increased accuracy requirement. A step length is considered too short when its value is smaller than a fixed quantity and it is smaller than a certain percentage of the step length computed at the previous outer iteration. The restart is performed only a time for each PR iteration. We also use an upper bound tol_{max} to prevent from too low accuracy requirements, based on the initial residual norm:

$$tol_{max} = 10^{-2}||r^0||.$$

This restarting CG strategy allows to use a reduced accuracy requirement in the computation of the directions, dinamically improving their "quality" when it is necessary, inside the current PR iteration. It has a very low computational cost, since it does not require a new factorization of the preconditioner, the matrices involved in the computation being invariate. Details of numerical results obtained by using the CG restarting strategy are shown in Tables 4.14, 4.15. We compare, at each PR iteration, the computed step length θ to the step length $\theta^{'}$ obtained by solving the KKT system with a direct method and we show the values of rejected step lengths in performing CG restarting. We also show the angle α between θ and $\theta^{'}$ and, when a CG restart is performed, the angle between $\theta^{'}$ and the rejected angle. We see that the CG restarting strategy is effective to compute "good" step length, since almost always the step lengths computed restarting CG iterations are better than the rejected steps and are not too different from that computed with a direct method.

In Table 4.16 the results obtained for all the considered QP problems by using PR-CPCG with the STOP2 criterion and restarting CG strategy are shown. We report the number of PR iterations and the total number of CG iterations. Further details are shown in the next tables, where the results concerning software comparisons are reported.

Problem	*STOP1*		*STOP2*	
	PR it	*CG it*	*PR it*	*CG it*
AUG3DCQP	15	16	15	15
CVXQP1-a	18	192	18	153
CVXQP1-b	22	290	21	223
CVXQP2-a	18	188	19	165
CVXQP2-b	21	287	21	217
CVXQP3-a	19	225	35	214
CVXQP3-b	23	371	24	321
GOULDQP3-a	8	2238	50	305
GOULDQP3-b	9	12167	44	9350
GOULDQP3-c	10	10363	50	264
MOSARQP1	19	70	38	76
MOSARQP2	15	52	30	57
STCQP2-a	24	195	50	78
STCQP2-a	30	305	30	237
STCQP2-a	33	365	33	287
RAND1	16	1028	50	790
RAND2	16	601	45	1079
RAND3	17	670	18	503
RAND4	17	1035	50	832
RAND5	18	716	20	617
RAND6	18	748	19	563
RAND7	19	837	32	871
RAND8	17	1276	50	826
RAND9	16	1392	50	890

Table 4.10: Number of PR iterations and CG iterations of PR-CPCG with STOP1 and STOP2 criteria.

Problem MOSARQP1, n = 2500, m = 700					
it	*CG it*	*obj. func.*	*step length θ*	*step length θ'*	α
1	1	4.1244314E+6	1.1673615E–2	3.0999504E–2	3.5217162E–3
2	1	4.1168257E+6	2.0194438E–3	5.1954647E–1	7.7426228E–3
3	1	4.1167489E+6	2.0443394E–5	5.3094238E–1	7.7162981E–3
4	1	4.1167481E+6	2.0445908E–7	5.3105709E–1	7.7160873E–3
5	1	4.1167481E+6	2.0445930E–9	5.3105824E–1	7.7160852E–3
6	1	4.1167481E+6	2.0447428E–11	5.3105825E–1	7.7160852E–3
7	1	4.1167481E+6	2.0351079E–13	5.3105825E–1	7.7160852E–3
8	1	4.1167481E+6	3.5703647E–15	5.3105825E–1	7.7160852E–3
9	1	2.3854301E+6	5.3141570E–1	4.0503318E–15	7.7160852E–3
10	1	1.0546949E+6	7.3932485E–1	7.4139100E–1	9.3765365E–3
11	1	1.0430482E+6	1.2102649E–2	3.7280249E–2	7.0122422E–3
12	1	1.0429314E+6	1.2233150E–4	3.8535204E–4	7.0073006E–3
13	1	1.0429303E+6	1.2234474E–6	3.8548177E–6	7.0072512E–3
14	1	1.0429302E+6	1.2234490E–8	3.8548315E–8	7.0072507E–3
15	1	1.0429302E+6	1.2234660E–10	3.8548852E–10	7.0072507E–3
16	1	1.0429302E+6	1.2244688E–12	3.8580446E–12	7.0072507E–3
17	1	1.0429302E+6	1.2167253E–14	3.8336466E–14	7.0072507E–3
18	1	1.0429302E+6	2.5173627E–15	7.9316826E–15	7.0072507E–3
19	1	1.0429302E+6	2.2965414E–15	7.2359209E–15	7.0072507E–3
20	1	1.0429302E+6	3.9747831E–16	1.2523709E–15	7.0072507E–3
21	1	1.0429302E+6	6.6246386E–16	2.0872849E–15	7.0072507E–3
22	1	1.7999737E+4	1.9133884E+0	1.8838657E+0	7.0072507E–3
23	1	4.3144364E+3	8.4014047E–1	9.0859128E–1	6.8379209E–3
24	1	–2.9858447E+2	7.3805723E–1	9.4106145E–1	6.6320512E–3
25	2	–2.5885914E+3	8.6669209E–1	8.6674547E–1	2.3058794E–3
26	3	–3.3564920E+3	8.1248735E–1	8.1247247E–1	7.7642026E–4
27	3	–3.6397346E+3	7.6539209E–1	7.6536730E–1	1.0740269E–3
28	3	–3.7606858E+3	8.1078182E–1	8.1356768E–1	1.1801662E–3
29	3	–3.8025719E+3	8.0942538E–1	8.0940495E–1	8.9222251E–4
30	3	–3.8155680E+3	7.8355826E–1	7.8353732E–1	8.4652787E–4
31	4	–3.8195450E+3	7.6001471E–1	7.6001482E–1	2.3127010E–5
32	3	–3.8207470E+3	7.2080015E–1	7.2081993E–1	1.8354064E–4
33	4	–3.8212162E+3	8.0369510E–1	8.0369405E–1	8.5561736E–6
34	4	–3.8213646E+3	8.7843958E–1	8.7843996E–1	1.5089381E–5
35	5	–3.8214007E+3	9.1275952E–1	9.1275952E–1	5.1619137E–8
36	5	–3.8214080E+3	9.1083740E–1	9.1083740E–1	5.1619137E–8
37	5	–3.8214095E+3	9.6406686E–1	9.6406686E–1	4.4703484E–8
38	5	–3.8214097E+3	9.0452763E–1	9.0452763E–1	1.0536712E–7

Table 4.11: Details of PR-CPCG with the STOP2 criterion for MOSARQP1.

Problem RAND8, n = 40000, m = 1000

it	*CG it*	*obj. func.*	*step length* θ	*step length* θ'	α
1	12	5.5890635E+4	2.3819130E–1	2.4004051E–1	4.3043098E–2
2	11	4.4598067E+4	2.9697747E–1	2.8216661E–1	6.2375555E–2
3	10	3.2810783E+4	4.0263040E–1	4.8400264E–1	6.1561997E–2
4	11	2.1803230E+4	5.3361583E–1	8.1342905E–1	6.3083858E–2
5	14	2.1604483E+4	1.4699823E–2	8.2719734E–1	5.4502048E–2
6	14	2.1602468E+4	1.5052344E–4	8.2306555E–1	5.4673317E–2
7	14	2.1602448E+4	1.5056091E–6	8.2302224E–1	5.4675133E–2
8	14	2.1602448E+4	1.5056126E–8	8.2302180E–1	5.4675151E–2
9	14	2.1602448E+4	1.5053765E–10	8.2302180E–1	5.4675151E–2
10	14	2.1602448E+4	1.5116627E–12	8.2302180E–1	5.4675151E–2
11	14	2.1602448E+4	4.1159132E–14	8.2302180E–1	5.4675151E–2
12	14	1.9656449E+4	1.4712104E–1	2.2563783E–14	5.4675151E–2
13	15	1.9627294E+4	2.4468146E–3	1.3190799E–1	4.8390748E–2
14	15	1.9627004E+4	2.4428646E–5	1.3338980E–1	4.8419585E–2
15	15	1.9627001E+4	2.4428252E–7	1.3340455E–1	4.8419874E–2
16	15	1.9627001E+4	2.4428507E–9	1.3340470E–1	4.8419877E–2
17	15	1.9627001E+4	2.4399782E–11	1.3340470E–1	4.8419877E–2
18	15	1.9627001E+4	2.1644672E–13	1.3340470E–1	4.8419877E–2
19	15	1.9627001E+4	4.0354473E–14	1.3340470E–1	4.8419877E–2
20	15	1.4094492E+4	4.8263121E–1	2.7509588E–14	4.8419877E–2
21	16	1.4012212E+4	1.0714588E–2	2.2730559E–1	5.3892380E–2
22	16	1.4011387E+4	1.0837115E–4	2.3493731E–1	5.4205712E–2
23	16	1.4011378E+4	1.0838363E–6	2.3501313E–1	5.4208882E–2
24	16	1.4011378E+4	1.0838378E–8	2.3501389E–1	5.4208914E–2
25	16	1.4011378E+4	1.0830126E–10	2.3501390E–1	5.4208914E–2
26	16	1.4011378E+4	1.2106358E–12	2.3501390E–1	5.4208915E–2
27	16	1.2412139E+4	2.1248967E–1	5.7311448E–14	5.4208915E–2
28	17	1.2388846E+4	3.6472929E–3	1.9421914E–1	5.3581436E–2
29	17	1.2388612E+4	3.6674497E–5	1.9586205E–1	5.3673457E–2
30	17	1.2388610E+4	3.6676531E–7	1.9587852E–1	5.3674381E–2

Table 4.12: Details of PR-CPCG with the STOP2 criterion for RAND8 (part 1).

Problem RAND8, n = 40000, m = 1000

it	*CG it*	*obj. func.*	*step length θ*	*step length θ'*	α
31	17	1.2388610E+4	3.6676055E–9	1.9587868E–1	5.3674391E–2
32	17	1.2388610E+4	3.6521222E–11	1.9587868E–1	5.3674391E–2
33	17	1.2388610E+4	4.5617985E–13	1.9587868E–1	5.3674391E–2
34	17	1.2388610E+4	2.6834109E–14	1.9587868E–1	5.3674391E–2
35	17	9.0122558E+3	5.4548940E–1	7.0612137E–14	5.3674391E–2
36	20	8.5345543E+3	1.2873839E–1	4.3056220E–1	6.3172072E–2
37	20	8.5299749E+3	1.3672932E–3	4.4110977E–1	7.0477624E–2
38	20	8.5299291E+3	1.3701421E–5	4.4122475E–1	7.0552360E–2
39	20	8.5299286E+3	1.3701830E–7	4.4122590E–1	7.0553109E–2
40	20	8.5299286E+3	1.3717293E–9	4.4122591E–1	7.0553116E–2
41	20	8.5299286E+3	1.3658011E–11	4.4122591E–1	7.0553116E–2
42	20	8.5299286E+3	6.3931116E–13	4.4122591E–1	7.0553116E–2
43	20	8.4045510E+3	3.7521482E–2	6.3855293E–15	7.0553116E–2
44	20	8.4032995E+3	3.8553073E–4	4.2294042E–3	7.2646656E–2
45	20	8.4032870E+3	3.8565801E–6	4.2714454E–3	7.2667897E–2
46	20	8.4032869E+3	3.8565959E–8	4.2718658E–3	7.2668110E–2
47	20	8.4032869E+3	3.8560028E–10	4.2718700E–3	7.2668112E–2
48	20	8.4032869E+3	3.9548746E–12	4.2718700E–3	7.2668112E–2
49	20	7.3870273E+3	3.1648662E–1	4.2718700E–3	7.2668112E–2
50	22	7.3728939E+3	5.8111587E–3	5.1385052E–2	8.1539647E–2

Table 4.13: Details of PR-CPCG with the STOP2 criterion for RAND8 (cont'd).

Problem MOSARQP1, n = 2500, m = 700							
it	*CG*	*obj. func.*	*step length*	*step length*	*rejected*	α	*rejected*
	it		θ	θ'	*step length*		*angle*
1	1	4.1244314E+6	1.1673615E–2	3.0999504E–2	—	3.5217162E–3	—
2	2	2.4231022E+6	5.1943896E–1	5.1954647E–1	2.0194438E–3	1.6158457E–3	7.7426228E–3
3	1	1.8306593E+5	1.6316047E+0	1.8715256E+0	—	1.2813857E–2	—
4	2	1.8923982E+4	1.4009245E+0	1.4482246E+0	—	3.3788702E–3	—
5	4	6.3287825E+3	7.1722754E–1	7.1581594E–1	6.7195648E–1	3.1484542E–5	1.7408407E–3
6	1	–9.9462105E+2	9.8760633E–1	9.8354721E–1	—	7.3733817E–3	—
7	2	–3.0725568E+3	9.6717991E–1	9.6711641E–1	—	2.0210771E–3	—
8	2	–3.5562010E+3	8.0497937E–1	8.0502570E–1	—	2.4982734E–3	—
9	3	–3.7442002E+3	8.6214440E–1	8.6213473E–1	—	6.5388398E–4	—
10	3	–3.7987887E+3	8.2723299E–1	8.2722333E–1	—	5.5469346E–4	—
11	3	–3.8138749E+3	7.5445625E–1	7.5445472E–1	—	1.6756441E–4	—
12	3	–3.8190284E+3	7.6301546E–1	7.6301835E–1	—	2.5433550E–4	—
13	3	–3.8206580E+3	7.6387417E–1	7.6387043E–1	—	2.4216331E–4	—
14	4	–3.8212051E+3	8.2455340E–1	8.2455373E–1	—	1.2285156E–5	—
15	4	–3.8213626E+3	8.8161352E–1	8.8161390E–1	—	1.5032635E–5	—
16	5	–3.8214006E+3	9.2184547E–1	9.2184547E–1	—	1.1827430E–7	—
17	5	–3.8214080E+3	9.1147490E–1	9.1147490E–1	—	7.8849534E–8	—
18	5	–3.8214095E+3	9.6326221E–1	9.6326221E–1	—	5.1619137E–8	—
19	5	–3.8214097E+3	9.0344500E–1	9.0344500E–1	—	1.0215721E–7	—

Table 4.14: Details of PR-CPCG with the STOP2 criterion with CG restart for MOSARQP1.

Problem RAND8, n = 40000, m = 1000

it	*CG*	*obj. func.*	*step length*	*step length*	*rejected*	α	*rejected*
	it		θ	θ'	*step length*		*angle*
1	12	5.5890635E+4	2.3819130E–1	2.4004051E–1	—	4.3043098E–2	—
2	11	4.4598067E+4	2.9697747E–1	2.8216661E–1	—	6.2375555E–2	—
3	10	3.2810783E+4	4.0263040E–1	4.8400264E–1	—	6.1561997E–2	—
4	11	2.1803230E+4	5.3361583E–1	8.1342905E–1	—	6.3083858E–2	—
5	27	1.1352583E+4	8.2725311E–1	8.2719734E–1	1.4699823E–2	5.9259064E–3	5.4502048E–2
6	35	6.7125626E+3	8.7189926E–1	8.7158534E–1	6.3257570E–1	6.1655298E–3	4.9394092E–2
7	46	5.0643049E+3	8.9587980E–1	8.9580967E–1	3.6698759E–1	4.0513660E–3	5.8922651E–2
8	55	4.6092515E+3	8.6030007E–1	8.5790487E–1	5.7613413E–1	1.5506299E–3	2.3604471E–2
9	43	4.4851293E+3	7.8396123E–1	9.5560734E–1	—	1.7853106E–2	—
10	51	4.4389504E+3	8.1865096E–1	8.1865557E–1	—	1.9307433E–2	—
11	66	4.4252145E+3	7.4711620E–1	9.3447192E–1	—	8.5871349E–3	—
12	76	4.4190342E+3	8.7814418E–1	8.7770015E–1	—	4.6380074E–3	—
13	81	4.4173089E+3	8.9580140E–1	9.9015182E–1	—	2.4687709E–3	—
14	86	4.4168374E+3	9.4827600E–1	1.0439692E+0	—	1.1825383E–3	—
15	90	4.4167384E+3	9.3117041E–1	9.3117298E–1	—	5.5523926E–4	—
16	87	4.4167147E+3	9.7996378E–1	9.7996463E–1	—	4.4187506E–4	—
17	89	4.4167100E+3	1.0345036E+0	1.0344989E+0	—	2.0041964E–4	—
18	94	4.4167094E+3	1.0336968E+0	1.0336964E+0	—	6.5953179E–5	—

Table 4.15: Details of PR-CPCG with the STOP2 criterion with CG restart for RAND8.

4.3.6 Numerical results: comparison with a direct approach for the KKT system

To better analyze the effects of CPCG on the PR algorithm, we compare the PR-CPCG implementation with stopping criterion for the CG steps based on (4.9)-(4.11) and CG restarting, to an implementation that was also developed applying a direct solver. This implementation, henceforth referred to as PR-DIR, performs a direct solution of the augmented system by using MA27 suite of routines from the Harwell Subroutine Library.

In Table 4.17 we report the number of PR iterations and the execution time, in seconds, of PR-CPCG and PR-DIR for the selected test problems. We also show the total number of CG iterations performed by PR-CPCG. More details on the time spent in the solution of the augmented system in both PR implementations, i.e. in the factorization of the preconditioner and in the execution of CPCG, as well as in the factorization of the augmented system matrix and in the solution of the triangular systems, are presented in Table 4.18. The total times required by the iterative and the direct system solution are also reported. Note that we were not able to run PR-DIR on two problems, due to the high memory requirements in the LDL^T factorization.

We see that the use of CPCG does not significantly affect the number of PR iterations. By comparing PR-CPCG and PR-DIR, we find that this number is the same for 55% of the problems; in the remaining cases, PR-CPCG generally requires one more iteration. The execution time of PR-CPCG is smaller then the time of PR-DIR for about 75% of the test cases. For most of them the time reduces by one to three orders of magnitude and such reduction is generally larger for larger problem sizes. This agrees with the observation that the execution time is dominated by the solution of the augmented system, and that the time required by the CPCG iterations is largely compensated by the time spent in the factorization of the preconditioner, which is often substantially smaller than the time for the factorization of the augmented system matrix. As expected, PR-CPCG requires a greater time than PR-DIR when the Hessian matrix is diagonal (AUG3DCQP) or “very close to being diagonal” (MOSARQP1 and MOSARQP2); in these cases, the difference between the times of the two PR implementations ranges approximately from 20% to 45%, where the percentage is computed with respect to the time of PR-DIR. Finally, PR-CPCG spends much more time than PR-DIR on the GOULDQP3 problems, since they require a very large number of CG iterations.

4.3.7 Numerical results: comparison with MOSEK

The aim of this subsection is to present numerical results of a comparison between the PRQP solver and MOSEK. We consider the PR-CPCG implementation with stopping criterion for the CG steps based on (4.9)-(4.11) and CG restarting. MOSEK was run setting the “relative gap termination tolerance” to 1E–7 and using the default values for all the other parameters (see [75]).

In Table 4.19, we show the number of iterations, the values of the primal objective functions and the relative duality gaps (defined as in (4.12)) at the computed solutions, the execution time (sec.) for PR-CPCG and MOSEK.

We see that PR-CPCG is faster than MOSEK in 70% of the test cases, although MOSEK performs less iterations. In particular, for most of the random problems, which are the largest ones, PR-CPCG requires an execution time which is from one to two orders of magnitude smaller. PR-CPCG is much faster on CVXQP2-b too. On the GOULDQP3 test cases the execution time of MOSEK is substantially smaller, according to the previous observation about the required number of CG iterations. As for PR-DIR, MOSEK is faster than PR-CPCG on AUG3DCQP, MOSARQP1 and MOSARQP2.

MOSEK is faster also on CVXQP3-b. Note that, for the six CVXQP test cases, given the size of the Hessian, the performance of PR-CPCG improves as the size of the contraint matrix decreases, since the time for factorizing and applying the preconditioner reduces.

Finally, we observe that the primal objective function values provided by PR-CPCG and MOSEK do not differ significantly. The objective function values of MOSEK, as well as the computed relative duality gaps, are often smaller, as expected since MOSEK implements an infeasible algorithm. For a few problems MOSEK does not satisfy the relative gap termination tolerance, reporting near-optimality for the computed solution.

Problem	*PR-CPCG* *STOP2 with CG restart*	
	PR it	*CG it*
AUG3DCQP	15	17
CVXQP1-a	18	160
CVXQP1-b	21	225
CVXQP2-a	18	153
CVXQP2-b	20	212
CVXQP3-a	19	185
CVXQP3-b	24	324
GOULDQP3-a	11	3460
GOULDQP3-b	10	5873
GOULDQP3-c	11	11300
MOSARQP1	19	58
MOSARQP2	15	45
STCQP2-a	25	162
STCQP2-b	30	246
STCQP2-c	33	292
RAND1	16	955
RAND2	18	531
RAND3	18	503
RAND4	17	774
RAND5	18	544
RAND6	18	533
RAND7	19	616
RAND8	18	970
RAND9	17	953

Table 4.16: Number of PR iterations and CG iterations of PR-CPCG with STOP2 criterion and CG restart.

Problem	*PR-CPCG*			*PR-DIR*	
	it	(*CG it*)	*time*	*it*	*time*
AUG3DCQP	15	(17)	2.51E–1	15	2.14E–1
CVXQP1-a	18	(160)	1.32E–1	18	1.14E+1
CVXQP1-b	21	(225)	1.19E+1	22	1.14E+4
CVXQP2-a	18	(153)	6.17E–2	18	2.48E+0
CVXQP2-b	20	(212)	9.11E–1	21	3.09E+3
CVXQP3-a	18	(185)	3.84E–1	19	3.13E+1
CVXQP3-b	24	(324)	8.34E+1	–	—
GOULDQP3-a	11	(3460)	1.42E+0	8	5.25E–2
GOULDQP3-b	10	(5873)	1.73E+1	8	2.69E–1
GOULDQP3-c	11	(11300)	7.60E+1	10	7.79E–1
MOSARQP1	19	(58)	2.43E–1	18	1.73E–1
MOSARQP2	15	(45)	1.79E–1	14	1.28E–1
STCQP2-a	25	(162)	1.01E–1	25	2.03E–1
STCQP2-b	30	(246)	7.14E–1	30	9.54E+0
STCQP2-c	33	(292)	1.99E+0	33	5.08E+1
RAND1	16	(955)	5.15E+0	16	3.49E+1
RAND2	18	(531)	3.21E+0	16	1.96E+2
RAND3	18	(503)	5.35E+0	17	1.53E+2
RAND4	17	(774)	8.07E+0	17	3.97E+2
RAND5	18	(544)	6.67E+0	18	8.59E+2
RAND6	18	(533)	7.50E+0	18	2.34E+3
RAND7	19	(616)	1.14E+1	–	—
RAND8	18	(970)	1.90E+1	17	1.30E+3
RAND9	17	(953)	3.07E+1	16	1.67E+3

Table 4.17: Number of PR iterations and execution time (sec.) of PR-CPCG and PR-DIR. The total number of CG iterations in PR-CPCG is also reported (in brackets).

Problem		*PR-PCG*			*PR-DIR*	
	prec.	*PCG*	*total*	*fact.*	*solve*	*total*
	time	*time*	*time*	*time*	*time*	*time*
AUG3DCQP	1.48E–1	3.91E–2	1.87E–1	1.45E–1	1.04E–2	1.55E–1
CVXQP1-a	6.05E–2	5.64E–2	1.17E–1	1.13E+1	5.59E–2	1.14E+1
CVXQP1-b	9.55E+0	2.11E+0	1.17E+1	1.14E+4	5.37E+0	1.14E+4
CVXQP2-a	1.34E–2	3.28E–2	4.63E–2	2.43E+0	2.87E–2	2.46E+0
CVXQP2-b	1.53E–1	5.98E–1	7.51E–1	3.09E+3	2.27E+0	3.09E+3
CVXQP3-a	2.33E–1	1.32E–1	3.65E–1	3.12E+1	1.01E–1	3.13E+1
CVXQP3-b	7.60E+1	7.11E+0	8.31E+1	—	—	—
GOULDQP3-a	1.98E–2	1.38E+0	1.40E+0	2.39E–2	1.68E–3	2.56E–2
GOULDQP3-b	1.13E–1	1.70E+1	1.71E+1	1.53E–1	1.19E–2	1.65E–1
GOULDQP3-c	2.90E–1	7.54E+1	7.57E+1	4.85E–1	3.46E–2	5.20E–1
MOSARQP1	1.47E–1	6.04E–2	2.07E–1	1.34E–1	9.17E–3	1.43E–1
MOSARQP2	1.05E–1	4.43E–2	1.49E–1	9.43E–2	6.62E–3	1.01E–1
STCQP2-a	2.66E–2	5.28E–2	7.94E–2	1.70E–1	7.22E–3	1.77E–1
STCQP2-b	1.48E–1	4.29E–1	5.77E–1	9.30E+0	9.08E–2	9.39E+0
STCQP2-c	3.78E–1	1.27E+0	1.65E+0	5.01E+1	2.51E–1	5.04E+1
RAND1	1.32E–1	4.76E+0	4.89E+0	3.45E+1	1.75E–1	3.46E+1
RAND2	1.63E–1	2.78E+0	2.94E+0	1.96E+2	4.00E–1	1.96E+2
RAND3	2.38E–1	4.70E+0	4.94E+0	1.52E+2	4.05E–1	1.52E+2
RAND4	2.48E–1	7.39E+0	7.64E+0	3.96E+2	6.47E–1	3.96E+2
RAND5	2.97E–1	5.92E+0	6.22E+0	8.57E+2	1.09E+0	8.58E+2
RAND6	3.90E–1	6.62E+0	7.01E+0	2.34E+3	2.01E+0	2.34E+3
RAND7	8.01E–1	1.01E+1	1.09E+1	—	—	—
RAND8	3.61E–1	1.79E+1	1.82E+1	1.30E+3	1.33E+0	1.30E+3
RAND9	4.43E–1	2.91E+1	2.95E+1	1.67E+3	1.61E+0	1.67E+3

Table 4.18: Time spent in the solution of the augmented system in PR-CPCG and PR-DIR.

Problem	*PR-PCG*				*MOSEK*			
	it	*obj. fun.*	*rel. gap*	*time*	*it*	*obj. fun.*	*rel. gap*	*time*
AUG3DCQP	15	–1.1437149E+3	5.81E–8	2.51E–1	9	–1.1437149E+3	1.42E–8	1.40E–1
CVXQP1-a	18	9.5628021E+5	5.70E–8	1.32E–1	13	9.5627955E+5	1.68E–8	3.80E–1
CVXQP1-b	21	9.5015317E+7	5.20E–8	1.19E+ 1	15	9.5014994E+7	4.45E–8	3.92E+1
CVXQP2-a	18	7.6457806E+5	5.00E–8	6.17E–2	13	7.6457807E+5	8.09E–8	3.90E–1
CVXQP2-b	20	7.6001729E+7	5.41E–8	9.11E–1	15	7.6001721E+7	7.75E–8	2.10E+2
CVXQP3-a	18	1.0437435E+6	9.65E–8	3.84E–1	14	1.0437447E+6	1.30E–6	4.70E–1
CVXQP3-b	24	1.0346454E+8	1.72E–8	8.34E+1	15	1.0346444E+8	7.26E–8	2.96E+1
GOULDQP3-a	11	–2.1031026E+3	1.12E–8	1.42E+0	4	–2.1031022E+3	3.46E–8	9.00E–2
GOULDQP3-b	10	–1.3279951E+4	6.43E–8	1.73E+1	5	–1.3279951E+4	1.13E–8	4.10E–1
GOULDQP3-c	11	–3.8149454E+4	4.56E–8	7.60E+1	6	–3.8149455E+4	4.00E–8	1.25E+0
MOSARQP1	19	–3.8214097E+3	3.00E–8	2.43E–1	10	–3.8214100E+3	6.33E–9	1.60E–1
MOSARQP2	15	–5.0525919E+3	4.45E–8	1.79E–1	8	–5.0525917E+3	8.45E–9	1.20E–1
STCQP2-a	25	1.0483316E+4	3.12E–8	1.01E–1	12	1.0483303E+4	3.05E–7	1.50E–1
STCQP2-b	30	6.2341714E+4	4.11E–8	7.14E–1	13	6.2341655E+4	5.94E–7	3.12E+0
STCQP2-c	33	1.4045189E+5	2.44E–8	1.99E+0	13	1.4045168E+5	1.83E–6	1.67E+1
RAND1	16	1.2018049E+3	6.25E–8	5.15E+0	9	1.2018040E+3	9.32E–8	1.38E+1
RAND2	18	1.9434943E+3	3.14E–8	3.21E+0	10	1.9434943E+3	2.80E–8	7.20E+1
RAND3	18	1.3889325E+3	2.11E–8	5.35E+0	10	1.3889324E+3	6.94E–8	4.96E+1
RAND4	17	3.1461022E+3	5.10E–8	8.07E+0	12	3.1461020E+3	7.63E–9	9.99E+1
RAND5	18	3.8319518E+3	3.89E–8	6.67E+0	11	3.8319522E+3	1.45E–8	6.12E+2
RAND6	18	5.9703612E+3	7.14E–8	7.50E+0	10	5.9703609E+3	8.61E–8	1.32E+3
RAND7	19	8.7214249E+3	2.55E–8	1.14E+1	10	8.7214249E+3	8.88E–8	2.21E+3
RAND8	18	4.4167094E+3	2.75E–8	1.90E+1	11	4.4167092E+3	9.62E–9	3.58E+2
RAND9	17	5.5535207E+3	3.80E–8	3.07E+1	10	5.5535209E+3	1.35E–8	5.50E+2

Table 4.19: Comparison of PR-CPCG and MOSEK.

List of Tables

Bibliography

[1] A. Altman and J. Gondzio, “Regularized symmetric indefinite systems in interior point methods for linear and quadratic optimization”, Optimization Methods and Software, 11-12, pp. 275–302, 1999.

[2] E.D. Andersen, J. Gondzio, C. Meszaros and X. Xu, “Implementation of Interior Point Methods for Large Scale Linear Programming”, in Interior Point Methods in Mathematical Programming, T. Terlaky ed., Kluwer Academic Publisher, pp. 189–252, 1996.

[3] E.D. Andersen and Y. Ye, “A Computational Study of the Homogeneous Algorithm for Large-Scale Convex Optimization”, Computational Optimization and Applications, 10 (3), pp. 243–269, 1998.

[4] O. Axelsson, “Preconditioning of indefinite problems by regularization”, SIAM Journal on Numerical Analysis, 16, pp. 58–69, 1979.

[5] V. Baryamureeba and T. Steihaug, “On the Convergence of an Inexact Primal-Dual Interior Point Method for Linear Programming”, Reports in Informatics No. 188, Department of Informatics, University of Bergen, Norway, 2000.

[6] S. Bellavia, “Inexact Interior Point method”, Journal of Optimization Theory and Applications, 96, pp. 109–121, 1998.

[7] H.Y. Benson, D. Shanno and R.J. Vanderbei, "Interior-point methods for nonconvex nonlinear programming: Jamming and numerical testing", Mathematical Programming, Ser. A, 99, pp. 35–48, 2004.

[8] M. Benzi, G. Golub and J. Liesen, "Numerical Solution of Saddle Point Problems", Acta Numerica, 14, pp. 1–137, 2005.

[9] L. Bergamaschi, J. Gondzio and G. Zilli, "Preconditioning Indefinite Systems in Interior Point Methods for Optimization", Computational Optimization and Applications, 28 (2), pp. 149–171, 2004.

[10] J.T. Betts, Practical methods for optimal control using nonlinear programming. Advances in Design and Control, SIAM, Philadelphia, 2001.

[11] R. Byrd, M.E. Hribar, and J. Nocedal, "An Interior Point Method for Large Scale Nonlinear Programming", SIAM Journal on Optimization, 9 (4), pp. 877–900, 1999.

[12] S. Cafieri, M. D'Apuzzo, M. Marino, A. Mucherino and G. Toraldo, "An Interior Point Solver for Large-Scale Quadratic Programming Problems with Bound Constraints", to appear in Journal of Optimization Theory and Applications, 129 (1), 2006.

[13] S. Cafieri, M. D'Apuzzo, V. De Simone, D. di Serafino, "On the Iterative Solution of KKT Systems in Potential Reduction Software for Large-scale Quadratic Problems", to appear in Computational Optimization and Applications.

[14] T.J. Carpenter and D.F. Shanno, "An interior point method for quadratic programs based on conjugate projected gradients", Computational Optimization and Applications, 2, pp. 5–28, 1993.

[15] A. Conn, N.I.M. Gould, D. Orban, Ph.L. Toint, “A primal-dual trust-region algorithm for minimizing a non-convex function subject to bound and linear equality constraints”, Mathematical Programming, 87 (2), 2000.

[16] J. Czyzyk, S. Mehrotra, M. Wagner, and S.J. Wright, “PCx User Guide (Version 1.1)”, Optimization Technology Center, Argonne National Laboratory, Tech. Rep. OTC 96/01, 1997.
See also `http://www-fp.mcs.anl.gov/otc/Tools/PCx/`

[17] M. D’Apuzzo and M. Marino, “Parallel Computational Issues of an Interior Point Method for Solving Large Bound Constrained Quadratic Programming Problems”, Parallel Computing, 29, pp. 467–483, 2003.

[18] I.I. Dikin, “Iterative solution of problems of linear and quadratic programming”, Soviet Mathematics Doklady, 8, pp. 674–675, 1967.

[19] H. Dollar, A. Wathen, “Incomplete factorization constraint preconditioners for saddle-point matrices”, Numerical Analysis Group, Oxford University, Research Rep. NA-04/01, 2004.

[20] J.J. Dongarra, J. Du Croz, I.S. Duff, S. Hammarling, “A set of Level 3 Basic Linear Algebra Subprograms”, ACM Transactions on Mathematical Software, 16 (1), pp. 1–17, 1990.

[21] I.S. Duff and J.K. Reid, “The Multifrontal Solution of Indefinite Sparse Symmetric Linear Equations”, ACM Transactions on Mathematical Software, 9 (3), pp. 302–325, 1983.

[22] C. Durazzi and V. Ruggiero, “Indefinitely Preconditioned Conjugate Gradient Method for Large Sparse Equality and Inequality Constrained Quadratic Problems”, Numerical Linear Algebra with Applications, 10 (8), pp. 673–688, 2003.

[23] ESOPO - Environment for Solving Optimization Problems Online: `http://www.esopo.unina2.it/`

[24] A.V. Fiacco, G.P. McCormick, “Nonlinear Programming: Sequential Unconstrained Minimization Techniques”, John Wiley and Sons, New York, 1968.

[25] R. Fletcher, Practical methods of optimization, Wiley, New York, Second Edition, 1987.

[26] A. Forsgren, P.E. Gill and J.R. Shinnerl, “Stability of symmetric ill-conditioned systems arising in interior methods for constrained optimization”, SIAM Journal on Matrix Analysis and Applications, 17, pp. 187–211, 1996.

[27] A. Forsgren, P.E. Gill, J.D. Griffin, “Iterative solution of augmented systems arising in interior methods”, UCSD Department of Mathematics Technical Report NA-05-03, 2005.

[28] R.W. Freund and F. Jarre, “A QMR-based interior-point algorithm for solving linear programs”, AT&T Bell Laboratories and Institute für Angewandte Mathematik und Statistik, Tech. Rep., 1995.

[29] R.M. Freund and S. Mizuno, “Interior Point Methods: Current Status and Future Directions”, in High Performance Optimization, H. Frenk et al. eds., Kluwer Academic Publishers, pp. 441–466, 2000.

[30] K.R. Frisch, “The logaritmic potential method of convex programming”, Technical Report, University Institute of Economics, Oslo, Norway, 1955.

[31] E.M. Gertz, J. Nocedal and A. Sartenaer, “A Starting-Point Strategy for Nonlinear Interior Methods,” Applied Math. Letters, 17, pp. 945–952, 2004.

[32] E.M. Gertz and S.J. Wright, “Object-Oriented Software for Quadratic Programming”, ACM Transactions on Mathematical Software, 29, pp. 58–81, 2003.
See also http://www.cs.wisc.edu/~swright/ooqp/

[33] P.E. Gill, W. Murray, D.B Ponceleon and M.A. Saunders, “Preconditioners for Indefinite Systems Arising in Optimization”, SIAM Journal on Matrix Analysis and Applications, 13, pp. 292–311, 1992.

[34] P.E. Gill, W. Murray, M.A. Saunders, J.A. Tomlin and M.H. Wright, “On projected Newton barrier methods for linear programming and an equivalence to Karmarkar’s projective method”, Mathematical Programming, 36, pp.183–209, 1986.

[35] P.E. Gill, W. Murray, and M.H. Wright, Practical optimization, Academic Press, London - New York 1981.

[36] D. Goldfarb, S. Liu, “An $O(n^3L)$ primal-dual potential reduction algorithm for solving convex quadratic programs”, Mathematical Programming, 61, pp. 161–170, 1993.

[37] G.H. Golub, C.F. Van Loan, Matrix Computations, Johns Hopkins University Press, third edition.

[38] G.H. Golub and A.J. Wathen, “An iteration for indefinite systems and its application to the Navier-Stokes equations”, SIAM Journal on Scientific Computing, 19, pp. 530–539, 1998.

[39] J. Gondzio, “HOPDM (version 2.12) - A Fast LP Solver Based on a Primal-Dual Interior Point Method”, European Journal of Operational Research, 85, pp. 221–225, 1995.
See also http://www.maths.ed.ac.uk/~gondzio/software/hopdm.html

[40] J. Gondzio and R. Sarkissian, "Parallel Interior Point Solver for Structured Linear Programs", Mathematical Programming, 96(3), pp. 561–584, 2003.
See also `http://www.maths.ed.ac.uk/~gondzio/parallel/solver.html`

[41] J. Gondzio and T. Terlaky, "A Computational View of Interior Point Methods for Linear Programming", in Advances in Linear and Integer Programming, J. Beasley ed., pp. 103–144, Oxford University Press, Oxford, England 1996.

[42] C.C. Gonzaga, "Polynomial affine algorithms for linear programming", Mathematical Programming, 49, pp. 7–21, 1990.

[43] C.C. Gonzaga, "Path-following methods for linear programming", SIAM Review, 34, pp. 167–224, 1992.

[44] N.I.M. Gould, M.E. Hribar and J. Nocedal, "On the Solution of Equality Constrained Quadratic Programming Problems Arising in Optimization", SIAM Journal on Scientific Computing, 23(4), pp. 1375–1394, 2001.

[45] N.I.M. Gould, D. Orban, Ph.L. Toint, "Numerical methods for large-scale nonlinear optimization", Acta Numerica, 14, pp. 299–361, 2005.

[46] N.I.M. Gould, D. Orban and Ph.L. Toint, "CUTEr (and SifDec): A Constrained and Unconstrained Testing Environment, revisited", ACM Transactions on Mathematical Software, 29 (4), pp. 373–394, 2003.

[47] C. Gueret, C. Prins and M. Sevaux, "Applications of optimization with Xpress-MP", Dash Optimization, 2002.
See also `http://www.dashoptimization.com/home/`

[48] C.G. Han, P.M. Pardalos and Y. Ye, “Computational Aspects of an Interior Point Algorithm for Quadratic Programming problems with Box Constraints”, in Large-Scale Numerical Optimization, T. Coleman and Y. Li eds., SIAM, 1990.

[49] J. Haws and C. Mayer, “Preconditioning KKT systems”, the Boeing Co., Tech. Rep. M&CT-Tech-01-021, 2001.

[50] IMB Editor, “ESSL for AIX V4.1 - Guide and References”, IBM Publications, 2003.

[51] CPLEX web page:
`http://www.ilog.com/products/cplex/index.cfm`

[52] M.T. Jones and P.E. Plassmann, “An improved incomplete Cholesky factorization”, ACM Trans. Math. Software, 21, pp. 5–17, 1995.

[53] N. Karmarkar, “A new polynomial-time algorithm for linear programming”, Combinatorica, 4, pp. 373–395, 1984.

[54] N. Karmarkar and K. Ramakrishnan, “Computational results of an interior point algorithm for large scale linear programming”, Mathematical Programming, 52, pp. 555–586, 1991.

[55] C. Keller, N. Gould, and A. Wathen, “Constraint preconditioning for indefinite linear systems”, SIAM Journal on Matrix Analysis and Application, 21(4), pp. 1300–1317, 2000.

[56] J. Kleinhans, G. Sigl, F. Johannes and K. Antreich, “GORDIAN: VLSI Placement by Quadratic Programming and Slicing Optimization”. IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, 10 (3), pp. 356–365, 1991.

[57] M. Kojima, N. Megiddo, T. Noma, A. Yoshise, "A unified approach to interior point algorithms for linear complementarity problems", Lecture Notes in Computer Science, Springer-Verlag, 1991.

[58] M. Kojima, S. Mizuno and A. Yoshise, "An $O(\sqrt{(}n)L)$ iteration potential reduction algorithm for linear complementarity problems", Mathematical Programming, 50, pp. 331–342, 1991.

[59] R.D. Lazarov and P.S. Vassilevski, "Preconditioning saddle-point problems arising from mixed finite element discretizations of elliptic equations", Numerical Linear Algebra with Applications, 3 (1), pp. 1–20, 1996.

[60] C.J. Lin, J.J. Moré, "Incomplete Cholesky factorizations with limited memory", SIAM Journal on Scientific Computing, 21, pp. 24–45, 1999.

[61] C.J. Lin, J.J. Moré, "Newton's Method for Large Bound-Constrained Optimization Problems", SIAM Journal on Optimization, 9 (4), pp. 1100–1127, 1999.

[62] L. Luksan and J. Vlcek, "Indefinitely preconditioned inexact Newton method for large sparse equality constrained nonlinear programming problems", Numerical Linear Algebra with Applications, 5, pp. 219–247, 1998.

[63] J. Lustig, R.E. Marsten, and D.F. Shanno, "On Implementing Mehrotra's Predictor Corrector Interior Point Method for Linear Programming", SIAM Journal on Optimization, 2, pp. 435–449, 1992.

[64] H.M. Markowitz, "Portfolio selection", Journal of Finance, 8, pp. 77–91, 1952.

[65] S. Mehrotra, "Implementations of affine scaling methods: Approximate solutions of systems of linear equations using preconditioned conjugate gradient method", ORSA Journal on Computing, 4 (2), pp. 102–118, 1992.

[66] S. Mehrotra and J. Wang, "Conjugate gradient based implementation of interior point methods for network flow problems", AMS Summer Conference Proceedings, SIAM, L. Adams and L. Nazareth eds., pp. 124–142, 1996.

[67] Cs. Mészáros, "The efficient implementation of interior point methods for linear programming and their applications", Eötvös Loránd University of Sciences, PhD thesis, 1996.
See also `http://www.sztaki.hu/~meszaros/bpmpd/`

[68] Cs. Mészáros, "The BPMPD interior point solver for convex quadratic problems", WP 98-8, Computer and Automation Research Institute, Hungarian Academy of Sciences, Budapest, 1998.

[69] S. Mizuno, M. Kojima and M.J. Todd, "Infeasible-interior-point primal-dual potential-reduction algorithms for linear programming", SIAM Journal on Optimization, 5, pp. 52–67, 1995.

[70] R.D.C. Monteiro and S.J. Wright, "A Globally and Superlinearly Convergent Potential Reduction Interior Point Method for Convex Programming", Technical Report 92–13, Department of Systems and Industrial Engineering, University of Arizona, Tucson, AZ 85721, 1992.

[71] J. Morales, J. Nocedal, R. Waltz, G. Liu, and J. Goux, "Assessing the Potential of Interior Methods for Nonlinear Optimization", in L.T. Biegler, O. Ghattas, M. Heinkenschloss and B van Bloemen Waanders, editors, Large-Scale PDE-Constrained Optimization, Springer-

Verlag, Lecture Notes in Computational Science and Engineering, 30, pp. 167–183, 2003.

[72] J.J More', G. Toraldo, "Algorithms for Bound Constrained Quadratic Programming Problems", Numerische Mathematik, 55, pp. 377–400, 1989.

[73] J.J More', G. Toraldo, "On the solution of Large Quadratic Programming Problems with Bound Constraints", SIAM Journal on Optimization, 1 (1), pp. 93–113, 1991.

[74] J.J. More' and S.J. Wright, Optimization Software Guide, Frontiers in Applied Mathematics, SIAM, Philadelphia, 1993.

[75] MOSEK ApS, The MOSEK optimization tools version 3.1 (Revision 28). Users manual and reference, 2002.
See also `http://www.mosek.com/`

[76] NEOS Server for Optimization:
`http://www-neos.mcs.anl.gov/`

[77] NEOS Guide for Optimization Software:
`http://www-fp.mcs.anl.gov/otc/Guide/SoftwareGuide/`

[78] Y. Nesterov and A. Nemirovskii, Interior point polynomial methods in Convex Programming, SIAM Series in Applied Mathematics, SIAM, 1994.

[79] J. Nocedal, S.J. Wright, Numerical Optimization, Springer Series in Operations Research, 1999.

[80] D.P. O'Leary, "Symbiosis between linear algebra and optimization", Journal of Computational and Applied Mathematics, 123, pp. 447–465, 2000.

[81] A.R.Oliveira and D.C.Sorensen, “Computational Experience with a Preconditioner for Interior Point Methods for Linear Programming”, Dept. of Computational and Applied Mathematics, Rice University, Houston, TX, Tech. Rep. TR97-28, 1997.

[82] P. Pardalos and H. Wolkowicz (eds.), Topics in Semidefinite and Interior-Point Methods, American Mathematical Society, 1998.

[83] I. Perugia and V. Simoncini, “Block-diagonal and Indefinite Symmetric Preconditioners for Mixed Finite Element Formulations”, Numerical Linear Algebra with Applications, 7 (7–8), pp. 585–616, 2000.

[84] L. Portugal, M. Resende, G. Veiga and J. Judice, “An efficient implementation of the infeasible primal-dual network flow method”, AT & T Bell Laboratories, New Jersey, Tech. Rep., 1994.

[85] F. Potra, S.J. Wright, “Interior Point methods”, Journal of Computational and Applied Mathematics, 124, (1–2), pp. 281–302, 2000.

[86] M. Rozloznik and V. Simoncini, “Krylov subspace Methods for Saddle Point Problems with Indefinite Preconditioning”, SIAM Journal on Matrix Analysis and Applications, 24(2), pp. 368–391, 2002.

[87] Y. Saad, Iterative Methods for Sparse Linear Systems - Second Edition, SIAM, Philadelphia, 2003.

[88] D.F. Shanno and R.J. Vanderbei, “Interior-points methods for nonconvex nonlinear programming: orderings and higher-order methods”, Mathematical Programming, 87 (2), pp. 303–316, 2000.

[89] K. Tanabe, “Centered Newton method for mathematical programming”, in System Modeling and Optimization: Proceedings of the 13th IFIP

Conference, M. Iri and K. Yajima eds., Lecture Notes in Control and Information Systems, 113, Springer-Verlag, pp. 197–206, 1988.

[90] M.J. Todd, "Potential-Reduction Methods in Mathematical Programming", Mathematical Programming, 76, pp. 3–45, 1996.

[91] M.J. Todd and Y.Ye, "A centered projective algorithm for linear programming", Mathematics of Operations Research, 15, pp. 508–529, 1990.

[92] R.H. Tutuncu, "An Infeasible-Interior-Point Potential-Reduction Method for Linear Programming" Mathematical Programming, 86 (2), pp. 313–334, 1999.

[93] R.J. Vanderbei, "LOQO User's Manual- Version 3.10", Department of Civil Enginreering and Operations Research, Princeton University, Tech. Rep. SOR-97-08, 1997.
See also `http://www.orfe.princeton.edu/~loqo/`

[94] R.J. Vanderbei, "Symmetric quasi-definite matrices", SIAM Journal on Optimization, 5, pp. 100–113, 1995.

[95] R.J. Vanderbei, Linear Programming: foundations and extensions, Kluwer Academic Publishers, second edition, 2001.

[96] R.A.Waltz, "KNITRO 4.0 User's Manual", Ziena Optimization, Inc., Evanston, IL, USA, 2004.
See also `http://www.ziena.com/knitro.html`

[97] M.H. Wright, "Interior Methods for Constrained Optimization", in Acta Numerica, 1, Cambridge University Press, pp. 341–407, 1992.

[98] M.H. Wright, “Ill-conditioning and computational error in primal-dual interior methods for nonlinear programming”, SIAM Journal on Optimization, 9, pp. 84–111, 1998.

[99] S.J. Wright, Primal-Dual Interior-Point Methods, SIAM, 1997.

[100] S.J. Wright, ”Stability of augmented system factorizations in interior point methods”, SIAM Journal on Matrix Analysis and Applications, 18, pp. 191–222, 1997.

[101] S.J. Wright, Optimization Software Packages, In Handbook of Applied Optimization, M. Resende and P. Pardalos, eds., Oxford University Press, pp. 1008–1015, 2002.

[102] S.J. Wright, “Interior Point Methods For Optimal Control Of Discrete-Time Systems”. Journal of Optimization Theory and Applications, 77, pp. 161–187,1993.

[103] Z. Yang, S. Areibi, “Global placement techniques for VLSI physical design automation”. In 15th International Conference on Computer Applications in Industry and Engineering, pp. 243–247, San Diego, California, 2002.

[104] Y. Ye, “An $O(n^3L)$ potential reduction algorithm for linear programming”, Mathematical Programming, 50, pp. 239–258, 1991.

[105] Y. Ye, Interior Point Algorithms: Theory and Analysis, John Wiley & Sons, 1997.