RELIABLE SOLUTION OF CONVEX QUADRATIC PROGRAMS WITH PARAMETRIC ACTIVE SET METHODS

A. POTSCHKA*, C. KIRCHES* , H. G. BOCK* , AND J. P. SCHLÖDER*

Abstract. Parametric Active Set Methods (PASM) are a relatively new class of methods to solve convex Quadratic Programming (QP) problems. They are based on tracing the solution along a linear homotopy between a QP with known solution and the QP to be solved. We explicitly identify numerical challenges in PASM and develop strategies to meet these challenges. To demonstrate the reliability improvements of the proposed strategies we have implemented a prototype code called rpasm. We compare the results of the code with those of other popular academic and commercial QP solvers on problems of a subset of the Maros-Mészáros test examples. Our code is the only one to solve all of the problems with the default settings. Furthermore, we propose how PASM can be used to compute local solutions of nonconvex QPs.

Key words. active set, convex, homotopy method, parametric quadratic programming

AMS subject classifications. 90C20, 90C25, 97N60, 97N80

1. Introduction. This paper is dedicated to the numerical solution of the convex quadratic programming (QP) problem

$$\min_{x\in\mathbb{R}^n} \frac{1}{2}x^{\mathrm{T}}Bx + b^{\mathrm{T}}x \quad \text{s. t.} \quad c^{\mathrm{l}} \le Cx \le c^{\mathrm{u}}, \tag{1.1}$$

with symmetric positive semidefinite Hessian matrix $B \in \mathbb{R}^{n\times n}$, constraint matrix $C \in \mathbb{R}^{m\times n}$, gradient vector $b \in \mathbb{R}^n$, and lower and upper constraint vectors $c^{\mathrm{l}}, c^{\mathrm{u}} \in \mathbb{R}^m$. Throughout this paper, inequality signs between vectors are to be understood elementwise.

The class of convex QP problems is important in its own right. Gould and Toint have been compiling a bibliography [19] of currently 979 publications which comprises many application papers from disciplines as diverse as portfolio analysis, structural analysis, VLSI design, discrete-time stabilization, optimal and fuzzy control, finite impluse response design, optimal power flow, economic dispatch, etc. Several benchmark and application problems are collected in a repository [25] which is accessible through the CUTEr testing environment [17]. Furthermore, QPs arise as subproblems in Sequential Quadratic Programming (SQP), a state-of-the-art method for the solution of nonlinear optimization problems.

1.1. Optimality conditions. For the characterization of solutions of QP (1.1) we partition the index set $\overline{m} = \{1, \dots, m\}$ into four disjoint sets

$$\mathcal{A}^{\mathrm{e}}(x) = \{i \in \overline{m} \,|\, c_i^{\mathrm{l}} = (Cx)_i = c_i^{\mathrm{u}}\}, \qquad \mathcal{A}^{\mathrm{l}}(x) = \{i \in \overline{m} \,|\, c_i^{\mathrm{l}} = (Cx)_i < c_i^{\mathrm{u}}\},$$
$$\mathcal{A}^{\mathrm{u}}(x) = \{i \in \overline{m} \,|\, c_i^{\mathrm{l}} < (Cx)_i = c_i^{\mathrm{u}}\}, \qquad \mathcal{A}^{\mathrm{f}}(x) = \{i \in \overline{m} \,|\, c_i^{\mathrm{l}} < (Cx)_i < c_i^{\mathrm{u}}\}$$

of equality, lower active, upper active, and free constraint indices, respectively. It is well known (see, e.g., [29]) that for any solution x^* of QP (1.1) there exists a vector

*Interdisciplinary Center for Scientific Computing, Heidelberg University, Im Neuenheimer Feld 368, 69120 Heidelberg, Germany (potschka@iwr.uni-heidelberg.de).

$y^* \in \mathbb{R}^m$ of so called Lagrange multipliers or dual variables such that

$$Bx^* + b - C^{\mathrm{T}} y^* = 0, \qquad c^{\mathrm{l}} \leq Cx^* \leq c^{\mathrm{u}}, \tag{1.2a}$$

$$(Cx^* - c^{\mathrm{l}})_i y_i^* = 0, \quad i \in \mathcal{A}^{\mathrm{l}}(x^*), \qquad y_i^* \geq 0, \quad i \in \mathcal{A}^{\mathrm{l}}(x^*), \tag{1.2b}$$

$$(Cx^* - c^{\mathrm{u}})_i y_i^* = 0, \quad i \in \mathcal{A}^{\mathrm{u}}(x^*), \qquad y_i^* \leq 0, \quad i \in \mathcal{A}^{\mathrm{u}}(x^*). \tag{1.2c}$$

Conversely, every primal-dual pair (x^*, y^*) which satisfies conditions (1.2) is a global solution of QP (1.1) due to semidefiniteness of the Hessian matrix B. The solution is unique if and only if the following two conditions are satisfied:

1. The active constraint rows $C_i, i \in \mathcal{A}^{\mathrm{e}} \cup \mathcal{A}^{\mathrm{l}} \cup \mathcal{A}^{\mathrm{u}}$, are linearly independent.
2. Matrix B is positive definite on the null space of the active constraints.

1.2. Existing methods. All existing methods for solving QPs are iterative and can be grossly divided into Active Set and Interior Point methods. Interior Point methods are sometimes called Barrier methods due to the possibility of different interpretations of the resulting subproblems (see, e.g., [29]). As a defining feature, Active Set methods keep a working set of active constraints and solve a sequence of equality constrained QPs. The working set must be updated between the iterates according to exchange rules which are based on conditions (1.2). In contrast, Interior Point methods do not use a working set but follow a nonlinear path, the so called central path, from a strictly feasible point towards the solution.

Active Set methods can be divided into primal, dual, and parametric methods, of which the primal Active Set method is the oldest and can be seen as a direct extension of the Simplex Algorithm [6]. Dual Active Set methods apply the primal Active Set method to the dual of QP (1.1) (which exists due to convexity). A relatively new variant of an Active Set method are Parametric Active Set Methods (PASM), e.g., the Parametric Quadratic Programming (PQP) method [3], which are the methods of interest in this paper. It is based on an affine-linear homotopy between a QP with known solution and the QP to be solved. It turns out that the optimal solutions depend piecewise affine-linear on the homotopy parameter and that along each affine-linear segment the active set is constant. The iterates of the method are simply the start points of each segment.

The numerical behaviour of Active Set and Interior Point methods is usually quite different: While Active Set methods need on average substantially more iterations than Interior Point methods, the numerical effort for one iteration is substantially less for Active Set methods. Often, one or the other method will perform favorably on a certain problem instance, underlining that both approaches are important.

We want to concisely compare the main advantages of the different Active Set versus Interior Point methods. One advantage of Interior Point methods is the regularizing effect of the central path which leads to well-defined behavior on problems with nonunique solutions due to, e.g., degeneracy or zero curvature in a feasible direction at the solution. An advantage of all Active Set methods is the possibility of hot starts which can give a substantial speed-up when solving a sequence of related QPs because the active set between the solutions usually changes only slightly. A unique advantage of PASM is that the so called Phase 1 is not needed. The term Phase 1 describes the solution of an auxiliary problem to find a feasible starting point for primal and dual Active Set methods or a strictly feasible starting point for Interior Point methods. The generation of an appropriate starting point with Phase 1 can be as expensive as the subsequent solution of the actual problem.

Code/Package	Interior Point	primal/dual Active Set	Parametric Active Set
BPMPD [27]	+		
BQPD [11]		+	
COPL_QP [35]	+		
CPLEX [20]	+	+	
CVXOPT [5]	+		
FICO$^{(TM)}$ Xpress Optimization Suite [8]	+	+	
GALAHAD [18]	+	+	
HOPDM [15]	+		
HSL [1]	+	+	
IQP [4]		+	
LOQO [32]	+		
MOSEK [2]	+		
OOQP [12]	+		
qpOASES [9]			+
rpasm (this paper)			+
QPOPT [13]		+	
QuadProg++ [7]		+	
quadprog [26]		+	
QuadProg [31]		+	

TABLE 1.1
Software for convex Quadratic Programming (in alphabetical order).

1.3. Existing software. The popularity of primal/dual Active Set and Interior Point methods is reflected in the large availability of free and commercial software products. A detailed list and comparison of the various implementations is beyond the scope of this paper. We want to restrict ourselves to citing a few implementations which we consider popular in Table 1.1. We further want to restrict our list to implementations which are specifically designed for QPs, although any Nonlinear Programming solver should be able to solve QPs.

The packages GALAHAD and FICO$^{(TM)}$ Xpress also provide the possibility of using crossover algorithms which start with an Interior Point method to eventually refine the solution by few steps of an Active Set method. CPLEX additionally offers the option of a concurrent optimizer which starts a Barrier and an Active Set method in parallel and returns the solution which was found in the shortest amount of CPU time.

For Parametric Active Set methods we are only aware of the code qpOASES [9, 10]. We have developed a prototype Matlab® code called rpasm to demonstrate the efficacy of the proposed techniques and strategies to fortify reliability of PASM.

1.4. Contribution. The contributions of this paper comprise the explicit identification of numerical challenges in PASM, the development of measures to meet these challenges, and the implementation of a prototype Matlab® code which demonstrates their efficacy on the Maros-Mészáros test examples [25] restricted to problems with up to 1000 variables and up to 1001 two-sided constraints (plus potential simple variable bounds).

1.5. Structure of the article. We start with recalling the Parametric Quadratic Programming (PQP) method [3] in §2 and identify its fundamental numerical challenges in §3. In §4 we develop strategies to meet these challenges. It follows a short description of the newly developed Matlab® code rpasm in §5 which we compare with other popular academic and commercial QP solvers in §6. We continue in §7 with a description of drawbacks of the reliability improving strategies. In §8 we discuss an extension to compute local minima of nonconvex QPs before we close the paper with conclusions and an outlook in §9.

2. Parametric Active Set methods.

2.1. The Parametric QP. The idea behind Parametric Active Set methods consists of following the optimal solutions on a homotopy path between two QP instances. Figuratively speaking, the homotopy morphs a QP with known solution into the QP to be solved. Let this homotopy be parametrized by $\tau \in [0,1]$. We then want to solve the one-parametric family of QP problems

$$\min_{x(\tau)\in\mathbb{R}^n} \frac{1}{2} x(\tau)^{\mathrm{T}} B x(\tau) + b(\tau)^{\mathrm{T}} x(\tau) \quad \text{s. t.} \quad c^{\mathrm{l}}(\tau) \le C x(\tau) \le c^{\mathrm{u}}(\tau), \tag{2.1}$$

with b, c^{l}, and c^{u} now being continuous functions $b(\tau), c^{\mathrm{l}}(\tau), c^{\mathrm{u}}(\tau)$. For fixed τ, let the optimal primal-dual solution be denoted by $z(\tau) = (x(\tau), y(\tau))$ which necessarily satisfies (1.2) (with $b = b(\tau), c^{\mathrm{l}} = c^{\mathrm{l}}(\tau), c^{\mathrm{u}} = c^{\mathrm{u}}(\tau)$). If we furthermore restrict the homotopy to affine-linear functions $b \in \mathcal{H}^n, c^{\mathrm{l}}, c^{\mathrm{u}} \in \mathcal{H}^m$, where

$$\mathcal{H}^k = \{ f : [0,1] \to \mathbb{R} \mid f(\tau) = (1-\tau) f(0) + \tau f(1), \quad \tau \in [0,1] \},$$

it turns out that the optimal solutions $z(\tau)$ depend piecewise linearly but not necessarily continuous on τ [3]. On each linear segment the active set is constant. Parametric Active Set algorithms follow $z(\tau)$ by jumping from one beginning of a segment to the next. We can immediately observe that this approach allows hot-starts in a natural way. As mentioned already in §1.2, no Phase 1 is needed to begin the method: We can always recede to the homotopy start $b(0) = 0, c^{\mathrm{l}}(0) = 0, c^{\mathrm{u}}(0) = 0, x(0) = 0, y(0) = 0$, although this is certainly not the best choice as we will discuss in §3 and §4.

2.2. The Parametric Quadratic Programming algorithm. A Parametric Active Set method was first described by Best [3] under the name Parametric Quadratic Programming (PQP) algorithm. Algorithm 1 displays the main steps. The lines preceded by a number deserve further explanation.

Step 1: Computation of step direction. The working set W is encoded as an m-vector with entries 0 or ± 1, where the i-th component indicates whether constraint i is inactive ($W_i = 0$), active at the lower bound ($W_i = -1$), or active at the upper bound ($W_i = +1$). Let C_W denote the matrix consisting of the rows C_i with $W_i \neq 0$ and let $c_W(\tau)$ denote a vector which consists of entries $c_i^{\mathrm{l}}(\tau)$ or $c_i^{\mathrm{u}}(\tau)$ depending on which (if any) bound is marked active in W_i. We can then determine the step direction $(\Delta x, \Delta y)$ by solving

$$K_W(\tau) \begin{pmatrix} \Delta x \\ -\Delta y_W \end{pmatrix} := \begin{pmatrix} B & C_W^{\mathrm{T}} \\ C_W & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ -\Delta y_W \end{pmatrix} = \begin{pmatrix} -(b(1) - b(\tau)) \\ c_W(1) - c_W(\tau) \end{pmatrix}. \tag{2.2}$$

The dual step Δy must be assembled from Δy_W by filling in zeros at the entries of constraints i which are not in the working set (i.e., $W_i = 0$). For the initial working set W we assume matrix C_W to have full rank and matrix B to be positive definite

Data: $B, C, b(1), c^{\mathrm{l}}(1), c^{\mathrm{u}}(1)$, and $z(0) = (x(0), y(0))$ optimal for $b(0), c^{\mathrm{l}}(0), c^{\mathrm{u}}(0)$ with working set $W \in \{-1, 0, 1\}^m$
Result: $z(1) = (x(1), y(1))$ or *infeasible* or *unbounded*
$\tau := 0$;
1 Compute step direction $\Delta z = (\Delta x, \Delta y)$ with current working set W;
2 Determine maximum homotopy step $\Delta\tau$;
if $\Delta\tau \geq 1 - \tau$ **then** return solution $z(1) := z(\tau) + (1 - \tau)\Delta z$;
Set $\tau^+ := \tau + \Delta\tau, z(\tau^+) := z(\tau) + \Delta\tau\Delta z$, and $W^+ := W$;
if *constraint l is blocking constraint* **then**
  Set $W_l^+ := \pm 1$;
3 Linear independence test for new working set W^+;
  if *linear dependent* **then**
4 Try to find exchange index k;
    if *not possible* **then** return *infeasible*;
5 Adjust dual variables $y(\tau^+)$;
    Set $W_k^+ := 0$;
  end
else *(sign change of k-th dual variable is blocking)*
  Set $W_k^+ := 0$;
6 Test for curvature of B on new working set W^+;
  if *nonpositive curvature* **then**
7 Try to find exchange index l;
    if *not possible* **then** return *unbounded*;
8 Adjust primal variables $x(\tau^+)$;
    Set $W_l^+ := \pm 1$;
  end
end
Set $\tau := \tau^+$ and $W := W^+$;
9 Possibly update matrix decompositions;
Continue with Step 1;

Algorithm 1: The Parametric Quadratic Programming Algorithm.

on the null space of C_W. Thus, matrix $K_W(0)$ is invertible. As we will see in Steps 3 and 6, the PQP algorithm ensures the full rank and positive definiteness properties and thus invertibility of $K_W(\tau)$ for all further steps through exchange rules for the working set W. We shall discuss a null space approach for the factorization of $K_W(\tau)$ in Step 9.

Step 2: Determination of step length. We can follow $z(\tau)$ in direction Δz along the current segment until either an inactive constraint becomes active (blocking constraint) or until the dual variable of a constraint in the working set becomes zero (blocking dual variable). Following the straight line with direction Δz beyond this point would lead to violation of conditions (1.2). The step length $\Delta\tau$ can be determined by *ratio tests*

$$\mathrm{RT} : \mathbb{R}^{m+m} \to \mathbb{R} \cup \{\infty\}, \quad \mathrm{RT}(u, v) = \min\{u_i / v_i \mid i \in \overline{m}, v_i > 0\}. \tag{2.3}$$

If the set of ratios is empty the minimum yields ∞ by convention. With the help of RT, the maximum step towards the first blocking constraint is given by

$$t_{\mathrm{p}} = \min\{\mathrm{RT}(Cx(\tau) - c^{\mathrm{l}}, -C\Delta x), \mathrm{RT}(c^{\mathrm{u}} - Cx(\tau), C\Delta x)\}, \tag{2.4}$$

and towards the first blocking dual variable by

$$t_{\mathrm{d}} = \mathrm{RT}(W \circ y(\tau), W \circ \Delta y), \tag{2.5}$$

where $\circ$ denotes elementwise multiplication to compensate for the opposite signs of the dual variables for lower and upper active constraints. The maximum step allowed is therefore

$$\Delta\tau = \min\{t_{\mathrm{p}}, t_{\mathrm{d}}\}.$$

Best [3] assumes that all occuring minimizations yield either ∞ or a unique minimizer with corresponding index $l \in \overline{m}$ if $\Delta\tau = t_{\mathrm{p}}$ or index $k \in \overline{m}$ if $\Delta\tau = t_{\mathrm{d}}$ from the sets of the ratio tests. The occurence of at least one nonunique minimizer is called a *tie*. We can distinguish between primal-dual ties if $t_{\mathrm{p}} = t_{\mathrm{d}}$, primal ties if l is not unique, and dual ties if k is not unique. In case of a tie it is not clear which constraint should be added or removed from the working set W and bad choices can lead to cycling or even stalling of the method. Thus, successful treatment of ties is paramount to the reliability of Parametric Active Set methods and shall be further discussed in §4.3.

Step 3: Linear independence test. The addition of a new constraint l to the working set W can lead to rank deficiency of C_{W^+} and thus loss of invertibility of matrix $K_W(\tau^+)$. The linear dependence of C_l on $C_i, i : W_i \neq 0$ can be verified by solving

$$\begin{pmatrix} B & C_W^{\mathrm{T}} \\ C_W & 0 \end{pmatrix} \begin{pmatrix} s \\ \xi_W \end{pmatrix} = \begin{pmatrix} C_l^{\mathrm{T}} \\ 0 \end{pmatrix}. \tag{2.6}$$

Only if $s = 0$ then C_l is linearly dependent on $C_i, i : W_i \neq 0$ [3]. The linear independence test can be evaluated cheaply by reusing the factorization needed to solve the step equation (2.2).

Step 4: Determination of exchange index k. It holds that $s = 0$. Let ξ be constructed from ξ_W like Δy from Δy_W. Equation (2.6) then yields

$$C_l = \sum_{i:W_i \neq 0} \xi_i C_i. \tag{2.7}$$

Multiplying equation (2.7) by λW_l^+ with $\lambda \geq 0$ and adding this as a special form of zero to the stationarity condition in equations (1.2) yields

$$\begin{aligned} &B(x(\tau) + \Delta\tau\Delta x) - b(\tau^+) \\ &\qquad = \sum_{i:W_i \neq 0} y_i(\tau^+) C_i^{\mathrm{T}} = -\lambda W_l^+ C_l^{\mathrm{T}} + \sum_{i:W_i \neq 0} (y_i(\tau^+) + \lambda W_l^+ \xi_i) C_i^{\mathrm{T}}. \end{aligned} \tag{2.8}$$

Thus, all coefficients of $C_i, i : W_i^+ \neq 0$ on the right hand side of equation (2.8) are also valid choices $\tilde{y}$ for the dual variables as long as they satisfy the sign condition $W_i^+ \tilde{y}_i \leq 0$. Hence we can compute the largest such λ with the ratio test

$$\lambda = \mathrm{RT}(-W_l^+(W \circ y(\tau^+)), W_l^+(W \circ \xi)). \tag{2.9}$$

If $\lambda = \infty$ then the parametric QP does not possess a feasible point beyond τ^+ and thus the QP to be solved (at $\tau = 1$) is infeasible. Otherwise, let k be a minimizing index of the ratio set.

Step 5: Jump in dual variables. Now let

$$\tilde{y}_i = \begin{cases} -\lambda W_i^+ & \text{for } i = l, \\ y_i(\tau^+) + \lambda W_i^+ \xi_i & \text{for } i : W_i \neq 0, \end{cases}$$

and set $y(\tau^+) := \tilde{y}$. It follows from construction of λ that $\tilde{y}_k = 0$ and thus, constraint k can leave the working set. As a consequence, matrix $C_{W^+\setminus\{k\}}$ preserves the full rank property and has the same null space as C_W, thus securing regularity of matrix $K_{W^+}(\tau^+)$.

Step 6: Curvature test. The removal of a constraint from the working set can lead to exposing directions of zero curvature on the null space of C_{W^+}, which is larger than the null space of C_W, leading to singularity of matrix $K_{W^+}(\tau^+)$. This can be detected by solving

$$\begin{pmatrix} B & C_W^{\mathrm{T}} \\ C_W & 0 \end{pmatrix} \begin{pmatrix} s \\ \xi_W \end{pmatrix} = \begin{pmatrix} 0 \\ -(e_k)_W \end{pmatrix}, \tag{2.10}$$

where e_k is the k-th column of the m-by-m identity matrix. Only if $\xi = 0$ then B is singular on the null space of C_{W^+} [3]. As for the linear independence test of Step 3, the curvature test can be evaluated cheaply by reusing the factorization needed to solve the step equation (2.2).

Step 7: Determination of exchange index l. It holds that $\xi = 0$ and s solves

$$Bs = 0, \quad C_k s = -1, \quad C_{W^+} s = 0. \tag{2.11}$$

Therefore all points $\tilde{x} = x(\tau^+) + \sigma s$ are also solutions if $\tilde{x}$ is feasible. We can compute the largest such $\sigma = \min\{\sigma^{\mathrm{l}}, \sigma^{\mathrm{u}}\}$ with the ratio tests

$$\sigma^{\mathrm{l}} = \mathrm{RT}(Cx(\tau^+) - c^{\mathrm{l}}, -Cs), \quad \sigma^{\mathrm{u}} = \mathrm{RT}(c^{\mathrm{u}} - Cx(\tau^+), Cs). \tag{2.12}$$

If $\sigma = \infty$ then the parametric QP is unbounded beyond τ^+, including the QP to be solved (at $\tau = 1$). Otherwise, let l be a minimizing index of a ratio set which delivers a final minimizer of σ.

Step 8: Jump in primal variables. Now set $x(\tau^+) := x(\tau^+) + \sigma s$. By construction of σ we have that either $C_l x(\tau^+) = c_l^{\mathrm{l}}$ (if $\sigma = \sigma^{\mathrm{l}}$) or $C_l x(\tau^+) = c_l^{\mathrm{u}}$ (otherwise). Thus l can be added to the working set via $W_l^+ := -1$ (if $\sigma = \sigma^{\mathrm{l}}$) or $W_l^+ := +1$.

Step 9: Update matrix decompositions. We summarize a null space approach for the solution of systems (2.2), (2.6), and (2.10). A range space approach is in general not possible if B is only semidefinite [29]. A direct factorization of $K_W(\tau)$ via $\mathrm{LDL}^{\mathrm{T}}$ factorization is also possible [24] but update formulae are in the general case not as efficient as for the null space approach. The alternative of iterative linear algebra methods for the indefinite matrix $K_W(\tau)$ are beyond the scope of this paper.

The null space approach is based on a QR decomposition of the transposed active constraint matrix

$$C_W^{\mathrm{T}} = Q\tilde{R} = \begin{pmatrix} Y & Z \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix} = YR, \quad Q^{\mathrm{T}}Q = \mathbb{I}.$$

Thus, the columns of Z constitute an orthonormal basis of the null space of C_W. The columns of Y are an orthonormal basis of the range space of C_W^{T} and the upper

triangular matrix R is invertible due to the full rank assumption on C_W. By assumption, the *projected Hessian* $Z^{\mathrm{T}}BZ$ is positive definite and lends itself to a Cholesky decomposition

$$Z^{\mathrm{T}}BZ = LL^{\mathrm{T}}$$

with invertible triangular matrix L. Exploiting $C_W Z = 0$ and $C_W Y = R$, the unitary basis transformation

$$\begin{pmatrix} Y & Z & 0 \\ 0 & 0 & \mathbb{I} \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} B & C_W^{\mathrm{T}} \\ C_W & 0 \end{pmatrix} \begin{pmatrix} Y & Z & 0 \\ 0 & 0 & \mathbb{I} \end{pmatrix} = \begin{pmatrix} Y^{\mathrm{T}}BY & Y^{\mathrm{T}}BZ & R \\ Z^{\mathrm{T}}BY & LL^{\mathrm{T}} & 0 \\ R^{\mathrm{T}} & 0 & 0 \end{pmatrix}$$

yields a block-triangular system which can be solved via backsubstitution. When the working set W changes by addition, removal, or substitution of constraints, the QR decomposition of C_{W^+} and following the Cholesky decomposition can be updated cheaply from the previous decompositions [14]. For exploitation of special structures of B and C in large scale applications we refer the interested reader to [21, 22].

This concludes our presentation of the Parametric Quadratic Programming algorithm.

2.3. Far bounds. Many applications lead to QPs where some of the constraint bounds $c_i^{\mathrm{l}}, c_j^{\mathrm{u}}, i \neq j$, are infinite to allow for one-sided constraints, e.g., $0 \leq x_i \leq \infty$. However, a homotopy from finite to infinite $c^{\mathrm{l}}(\tau)$ and $c^{\mathrm{u}}(\tau)$ is not possible. The flipping bounds strategy that shall be described in §4.1 relies on finiteness of $c^{\mathrm{l}}(\tau)$ and $c^{\mathrm{u}}(\tau)$. We circumvent this problem by application of a so-called *far bounds* strategy. It is based on the following idea: Let $M > 0$ be given. If M is large enough then a solution (x, y) of (1.1) is also a solution of

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} x^{\mathrm{T}} B x + b^{\mathrm{T}} x \quad \text{s. t.} \quad \tilde{c}^{\mathrm{l}} \leq Cx \leq \tilde{c}^{\mathrm{u}}, \tag{2.13}$$

where $\tilde{c}_i^{\mathrm{l}} = \max(c_i^{\mathrm{l}}, -M), \tilde{c}_i^{\mathrm{u}} = \min(c_i^{\mathrm{u}}, M), i = 1, \ldots, m$. We call a constraint bound which attains the value $\pm M$ a far bound. Algorithmically we solve a sequence of QPs with growing far bounds value M, see Algorithm 2. The total solution time will mostly be dominated by the solution of the first QP as consecutive QPs can be efficiently warm started.

```
Initialize M = 10^3;
repeat
    Solve QP (2.13);
    if no far bounds active then return QP solution;
    Grow far bounds: M := 10^3 M;
until M > 10^20 ;
if last QP infeasible then return QP infeasible;
else return QP unbounded;
```

Algorithm 2: The far bounds strategy.

3. Fundamental numerical challenges. In this section we describe the numerical challenges that occur in the PQP algorithm. We shall develop countermeasures in §4.

One fundamental challenge in many applications is ill-posedness of problems: Small changes in the data of the problem lead to large changes in the solution. This challenge necessarily propagates through the algorithm and leads to ill-conditioned matrices K_W. As a consequence the results of the step computation (2.2), the linear independence test (2.6), and the curvature test (2.10) can be errorneous up to $\mathrm{cond}(K_W)$ times machine precision in relative error [34]. This, in turn, can lead to very instable ratio tests (2.3) and wrong choices for the working set which can cause the algorithm to break down.

Rounding errors can also accumulate over several iterations and lead to the parametric "solution" $z(\tau)$ being optimal with an accuracy much less than machine precision. This phenomenon is called *drift*. Large drift can also lead to breakdown of the algorithm because the general assumption of optimality of $z(\tau)$ is violated.

Furthermore, the termination criterion must also be adapted to work reliably on both well- and ill-conditioned problems.

Ill-conditioning can also be introduced if the null space of C_W captures two eigenvalues of B with high ratio, leading to ill-conditioning of the Cholesky factors L. In the extreme case, the next step for the dual variables can be afflicted with a large error, causing again instability in the ratio tests.

The second fundamental challenge is the occurence of comparisons with zero, a delicate subject in the presence of rounding errors. These comparisons permeate the algorithm from the sign condition in the ratio tests (2.3) to the tests for linear dependence (2.6) or zero curvature (2.10).

The third fundamental challenge is the treatment of ties, i.e., nonuniqueness of minimizers of the ratio tests (2.3). Consider the case mentioned in §2.1 of a homotopy starting at $x(0) = 0, y(0) = 0, b(0) = 0, c^{\mathrm{l}}(0) = 0, c^{\mathrm{u}}(0) = 0, W = 0$. Clearly $(x(0), y(0))$ is an optimal solution at $\tau = 0$ regardless of the choice of B and C. Note that for the PQP algorithm the choice of $W = 0$ is only possible if B is positive definite. The first step direction will then point towards the unconstrained minimizer of the objective. If more than one constraint is active in the solution at $\tau = 1$ then the primal ratio test (2.4) for determination of the step length yields a (multiple) primal tie with $\Delta\tau = t^{\mathrm{p}} = 0$. Of all possible ratio test minimizers, one has to be chosen. One approach seems to be to employ pricing heuristics from primal/dual Active Set methods but we prefer a different approach which we discuss in §4.3. In the following iterations, primal-dual ties can occur while still $\Delta\tau = 0$. Thus *cycling*, the repeated addition and removal of the same constraints without any progress, can be possible which leads to stalling of the method. We are not aware of any pricing strategy which can avoid the problem of cycling. Ties also occur naturally in the case of degenerate QPs, where the optimal primal or dual variables are not uniquely determined.

4. Strategies to meet numerical challenges. We propose to employ the following strategies for Parametric Active Set methods to meet the fundamental numerical challenges described in §3.

4.1. Rounding errors and ill-conditioning. The most effective countermeasure against the challenges of ill-conditioning is to iteratively improve the quality of the linear system solutions via

Iterative Refinement [34]. We have already mentioned that the relative error in the solution z of

$$K_W z = d$$

can be as high as $\mathrm{cond}(K_W)$ times machine precision. Thus, if $\mathrm{cond}(K_W) \approx 10^{10}$ and we perform computations in double precision, the solution z can have as little as six valid decimal digits. Iterative Refinement

$$z^0 = 0, \quad r^k = K_W z^k - d, \quad K_W \delta z^k = r^k, \quad z^{k+1} = z^k - \delta z^k$$

recovers a fixed number of extra valid digits in each iteration. In the previous example, the iterate z^2 has at least twelve valid decimal digits after only one extra step of Iterative Refinement. It is worth noticing that compared to the original "solution" z^1 each iteration only needs to perform one additional matrix-vector-multiplication with K_W and one backwards solve with the decomposition of K_W described in §2.2. In exact arithmetic $z^{k+1} = z^k$ for all $k \geq 1$.

Drift correction. A very effective strategy to avoid drift can be formulated if the PQP algorithm is cast in a slightly different framework. After each iteration, we rescale the homotopy parameter to $\tau = 0$, thus interpreting the iterate $z(\tau^+)$ as a new starting value $z(0)$. This does not avoid drift yet but allows for modifications to restore consistency of the starting point via

$$\begin{aligned}
c_i^{\mathrm{l}}(0) &:= \begin{cases} C_i x(0) & \text{if } W_i = -1, \\ \min\{c_i^{\mathrm{l}}(0), C_i x(0)\} & \text{otherwise,} \end{cases} \\
c_i^{\mathrm{u}}(0) &:= \begin{cases} C_i x(0) & \text{if } W_i = +1, \\ \max\{c_i^{\mathrm{u}}(0), C_i x(0)\} & \text{otherwise,} \end{cases} \\
y_i(0) &:= \begin{cases} \max\{0, y_i(0)\} & \text{if } W_i = -1, \\ \min\{0, y_i(0)\} & \text{if } W_i = +1, \\ 0, & \text{otherwise,} \end{cases} \\
b(0) &:= Bx(0) - C^{\mathrm{T}} y(0),
\end{aligned}$$

for $i \in \overline{m}$. This annihilates the effects of drift after every iteration, at the cost of splitting up the single homotopy into a sequence of homotopies which are, however, very close to the remaining part of the original homotopy. In exact arithmetic the proposed modification does not alter any value.

Termination Criterion. It is tempting to use the homotopy parameter τ in the termination criterion as proposed in Algorithm 1. However, this choice renders the termination criterion dependent on the choice of the homotopy start, an undesirable property. Instead, we propose to use the relative distance δ in the data space

$$\begin{aligned}
&\Delta^0 = (b(\tau), c^{\mathrm{l}}(\tau), c^{\mathrm{u}}(\tau)), \quad s_j = (\Delta_j^1 - \Delta_j^0)/\max\{1, |\Delta_j^1|\}, \quad j = 1, \ldots, n+m+m, \\
&\Delta^1 = (b(1), c^{\mathrm{l}}(1), c^{\mathrm{u}}(1)), \qquad \delta = \|s\|_\infty.
\end{aligned}$$

This choice renders the termination criterion independent of the condition number of $K_W(1)$. We observe that the termination criterion can give no guarantee for the distance to the exact solution. Instead, a backwards analysis result holds: The computed solution is the exact solution to a problem which is as close as δ to the one to be solved. The numerical results presented in §6 were obtained with the tolerance $\delta \leq \delta_{\mathrm{term}} = 1\mathrm{e}7\,\mathrm{eps}$.

Ill-conditioning of the Cholesky factors L. To avoid ill-conditioning of the Cholesky factors L we have developed the so-called *flipping bounds* strategy. Flipping bounds is similar to taking long steps in the dual Simplex method [23, 30], where one variable

changes in the working set from upper to lower bound immediately without becoming inactive in between, i.e., it *flips*. Flipping is only possible if $c^{\mathrm{l}}(1)$ and $c^{\mathrm{u}}(1)$ have only finite entries, which is guaranteed by the far bounds strategy described in §2.3. We modify the PQP algorithm in the following way: If a constraint l was removed without requiring another constraint k to enter the active set, we monitor the size of the smallest entry ℓ_i of the diagonal of L in Step 9. If $\ell_i^2 < \delta_{\mathrm{curv}}$ we have detected a small eigenvalue in L which corresponds to a small eigenvalue of B now uncovered through the removal of constraint l. To avoid ill-conditioning of LL^{T}, we introduce a jump in the QP homotopy by requiring that the other bound of constraint l is moved such that it becomes active immediately (hence the name flipping bounds) through setting

$$\begin{aligned} \tilde{c}_l^{\mathrm{l}}(\tau^+) &:= c_l^{\mathrm{u}}(\tau^+), W_l^+ = -1, && \text{if } W_l = +1, \\ \tilde{c}_l^{\mathrm{u}}(\tau^+) &:= c_l^{\mathrm{l}}(\tau^+), W_l^+ = +1, && \text{if } W_l = -1. \end{aligned}$$

The entries $j \neq l$ of $\tilde{c}^{\mathrm{l}}$ and $\tilde{c}^{\mathrm{u}}$ are set to the corresponding entries of c^{l} and c^{u}. Consequently the Cholesky decomposition from the previous step stays valid for the current projected Hessian.

The numerical results presented in §6 were obtained with the curvature tolerance $\delta_{\mathrm{curv}} = 1\mathrm{e}4\,\mathrm{eps}$.

4.2. Comparison with zero.

Ratio tests. In the ideal ratio test (2.3) we take a minimum over a subset of m quotients with strictly positive denominator. The presence of round-off error makes it necessary to substitute the ideal ratio test by an expression with adjustable tolerances, e.g.,

$$\begin{aligned} u_i^{\mathrm{cut}} &= \max(u_i, \varepsilon_{\mathrm{cut}}), \quad i \in \overline{m}, \\ \mathrm{RT}_{\mathrm{r}}(u, v, \varepsilon_{\mathrm{cut}}, \varepsilon_{\mathrm{den}}, \varepsilon_{\mathrm{num}}) &= \min\{u_i^{\mathrm{cut}}/v_i \mid i \in \overline{m}, v_i \geq \varepsilon_{\mathrm{den}}, u_i^{\mathrm{cut}} \geq \varepsilon_{\mathrm{num}}\}. \end{aligned}$$

We now explain the purpose of the three tolerances: The *denominator tolerance* $\varepsilon_{\mathrm{den}} > 0$ describes which small but positive values of v_i should already be considered less than or equal to zero. They are consequently discarded as candidates for the minimum.

The *cutting tolerance* $\varepsilon_{\mathrm{cut}}$ and the *numerator tolerance* $\varepsilon_{\mathrm{num}}$ offer the freedom of two different treatments for numerators close to zero. If $\varepsilon_{\mathrm{cut}} > \varepsilon_{\mathrm{num}}$ then negative numerators are simply cut off at $\varepsilon_{\mathrm{cut}}$ before the quotients are taken, yielding that the minimum is greater or equal to $\varepsilon_{\mathrm{cut}}/\varepsilon_{\mathrm{den}}$. For instance, we set $\varepsilon_{\mathrm{cut}} = 0$ in the ratio tests for determination of the step length (2.4) and (2.5). This choice is motivated by the fact that in exact arithmetic $u_i \geq 0$ for all $i \in \overline{m}$ with $v_i > 0$. Thus, only values u_i which are negative due to round-off are manipulated and the step length satisfies $\Delta\tau \geq 0$ also in finite precision arithmetic.

If $\varepsilon_{\mathrm{cut}} \leq \varepsilon_{\mathrm{num}}$ then cutting does not have any effect. We have found it beneficial for the reliability of PASM to set $\varepsilon_{\mathrm{num}} = \varepsilon_{\mathrm{den}}$ in the ratio tests (2.9) and (2.12) for finding exchange indices.

The numerical results presented in §6 were obtained with the ratio test tolerances $\varepsilon_{\mathrm{den}} = -\varepsilon_{\mathrm{num}} = 1\mathrm{e}3\,\mathrm{eps}$ and $\varepsilon_{\mathrm{cut}} = 0$ for step length determination and $\varepsilon_{\mathrm{den}} = \varepsilon_{\mathrm{num}} = -\varepsilon_{\mathrm{cut}} = 1\mathrm{e}3\,\mathrm{eps}$ for the remaining ratio tests.

Linear independence and zero curvature test. After solution of systems (2.6) and (2.10) for s and ξ_W we must compare the norm of s or ξ with zero. Let $\zeta = (s, \xi)$.

We propose to use the relative conditions

$$\|s\|_\infty \le \varepsilon_{\text{test}} \|\zeta\|_\infty \qquad \text{for the linear dependence test and} \tag{4.1}$$

$$\|\xi\|_\infty \le \varepsilon_{\text{test}} \|\zeta\|_\infty \qquad \text{for the zero curvature test.} \tag{4.2}$$

We remark that $\|\zeta\|_\infty = \|\xi\|_\infty$ if $s = 0$ and $\|\zeta\|_\infty = \|s\|_\infty$ if $\xi = 0$. Thus we can replace $\|\zeta\|_\infty$ in the code by $\|\xi\|_\infty$ in test (4.1) and by $\|s\|_\infty$ in test (4.2).

The numerical results presented in §6 were obtained with $\varepsilon_{\text{test}} = 1\text{e}5\,\text{eps}$.

4.3. Cycling and ties. Once ties have occured, their resolution is a costly affair because of the combinatorial nature of the decision which subset of the possible constraints should be chosen to leave or enter the working set. This decision can be based on the solution of yet another QP of larger size then the original problem [33] or on heuristics similar to anti-cycling rules in the Simplex method.

We prefer a different approach instead. The idea behind the strategy we propose for ties is simple: Instead of trying to treat ties, we try to avoid them in the first place. The strategy is as simple as the idea and exploits the homotopy framework of PASM. Let a homotopy start $b(0), c^{\text{l}}(0), c^{\text{u}}(0)$ with optimal solution $(x(0), y(0))$ and working set W be given. Then, for every triple of m-vectors $r^0, r^1, r^2 \ge 0$, the primal-dual pair $(x(0), \tilde{y}(0))$ with

$$\tilde{y}_i(0) = \begin{cases} y_i(0) + r_i & \text{if } W_i = -1, \\ y_i(0) & \text{if } W_i = 0, \\ y_i(0) - r_i & \text{if } W_i = +1, \end{cases} \qquad i \in \overline{m},$$

is an optimal solution to the homotopy start $\tilde{b}(0), \tilde{c}^{\text{l}}(0), \tilde{c}^{\text{u}}(0)$, where for $i \in \overline{m}$

$$\tilde{c}^{\text{l}}_i(0) = \begin{cases} c^{\text{l}}_i(0), & \text{if } W_i = -1, \\ c^{\text{l}}_i(0) - r^1_i, & \text{otherwise,} \end{cases}$$

$$\tilde{c}^{\text{u}}_i(0) = \begin{cases} c^{\text{u}}_i(0), & \text{if } W_i = +1, \\ c^{\text{u}}_i(0) + r^2_i, & \text{otherwise,} \end{cases}$$

$$\tilde{b}(0) = -(Bx(0) - C^{\text{T}}\tilde{y}(0)).$$

In other words, if we move the inactive constraint bounds further away from $Cx(0)$ and the dual variables of the active constraints further away from zero, $x(0)$ stays feasible and $b(0)$ can be adapted to restore optimality of $(x(0), \tilde{y}(0))$ with the same working set W. Recall that the ratio tests depend exactly on the residuals of the inactive constraints and the dual variables of the active constraints. In our numerical tests, the simple choice of

$$r^j_i = (1 + (i-1)/(m-1))/2, \quad j = 0, 1, 2, \quad i \in \overline{m},$$

has proven to work reliably. Because of the shape of r^j, we call this strategy *ramping*. It is important to avoid two entries of r^j to have the same value because many problems have special structures, e.g., variable bounds of the same value for several variables which lead to primal ties if the homotopy starts with the same value for each of these variables. Of course, the choice of linear ramping is somewhat arbitrary and if a problem happens to have variable bounds in the form of a ramp, ties are again possible. However, this kind of structure is much less common than equal variable bounds.

We employ ramping in the starting point of the homotopy and also after an iteration which resulted in a zero step $\Delta\tau = 0$. Of course, this can lead to large jumps in the problem homotopy and practically catapult the current $b(0) := \tilde{b}(0)$ away from $b(1)$. However, a PASM is capable of reducing even a large distance in the data space to zero in one step, provided the active set is right. Thus, the distance of the working set W to the active set of the solution is a more appropriate measure of the progress of a PASM. By construction, the active set is preserved by the ramping strategy.

We further want to remark that ties can never be completely avoided. For instance in case of a QP whose solution lies in a degenerate corner, a tie must occur in (at least) one iteration of a PASM. In the numerical examples we have treated so far, the ramping strategy effectively deferred these ties to the final step, where a tie is not a problem any more because the solution at the end of the last homotopy segment is already one of infinitely many solutions of the QP to be solved and no ties must be resolved in the solution.

5. The code rpasm: A PASM in Matlab®. We have implemented the strategies proposed in §4 in a Matlab® code called rpasm. The main purpose of the code is to demonstrate reliability and solution quality on the test set. In favor of code simplicity we have refrained from employing special structure exploiting linear algebra routines which could greatly enhance the runtime of the code. The three main features in the C++ PASM code qpOASES [9, 10] for runtime improvement in the linear algebra routines are special treatment of variable bounds, updates for QR decompositions, and appropriate updates for Cholesky decompositions. Of the three, rpasm only performs QR updates. Variable bounds are simply treated as general inequality constraints. Cholesky decompositions are computed from scratch after a change in the active set. Furthermore, only dense linear algebra routines are used which do not exploit the often sparse matrices in the test examples. Another feature which is common in most commercial QP solvers is the use of a preprocessing/presolve step to reduce the problem size by eliminating fixed variables and dispensable constraints and possibly scaling the data. We will see that rpasm works reliably even without preprocessing.

The merging of the features of qpOASES and rpasm exceeds the aim of this paper and will be presented in a future publication.

6. Comparison with existing software. From the codes contained in Table 1.1 we use the ones which are freely available for academic purposes and come with a Matlab® interface, i.e., CPLEX, OOQP, qpOASES, plus the Matlab® solver quadprog and the newly developed rpasm. The programs cover the range of Primal Active Set (CPLEXP, quadprog), Dual Active Set (CPLEXD), Barrier/Interior Point (CPLEXB, OOQP), and Parametric Active Set (qpOASES, rpasm). For rpasm, we further differentiate between a version without iterative refinement (rpasm0) and with one possible step of iterative refinement (rpasm1). All codes were used with their default settings on all problems.

6.1. Criteria for comparison. We compare the runtime and the quality of the solution. Runtime was measured as the average runtime of three runs on one core of a Intel® Core™ i7 with 2.67 GHz and 8 MB cache in Matlab® 7.6 under Linux 2.6 (64 bit). The quality of solutions (x^*, y^*) was measured using a residual ρ of

conditions (1.2) defined via

$$\rho_{\text{stat}} = \|Bx^* + b - C^{\mathrm{T}} y^*\|_\infty, \qquad \rho^{\text{l}}_{\text{cmpl}} = \max\{|(Cx^* - c^{\text{l}})_i y_i^*| \mid y_i^* \geq +10\,\text{eps}\},$$
$$\rho_{\text{feas}} = \max(0, c^{\text{l}} - Cx^*, Cx^* - c^{\text{u}}), \qquad \rho^{\text{u}}_{\text{cmpl}} = \max\{|(Cx^* - c^{\text{u}})_i y_i^*| \mid y_i^* \leq -10\,\text{eps}\},$$
$$\rho = \max(\rho_{\text{stat}}, \rho_{\text{feas}}, \rho^{\text{l}}_{\text{cmpl}}, \rho^{\text{u}}_{\text{cmpl}}).$$

In Table 8.1 we have assembled the results for problems from the Maros-Mészáros test set [25] with at most $n = 1000$ variables and $m = 1001$ two-sided inequality constraints (not counting variable bound constraints). We have visualized the results in the performance graphs of Figures 6.1 and 6.2. The graphs display a time factor on the abscissa versus the percentage of problems that each code was able to solve within the time factor times the runtime of the fastest method for each problem. Roughly speaking, the graph of a fast method is close to the left hand side of the diagram, the graph of a reliable method is close to the top of the diagram.

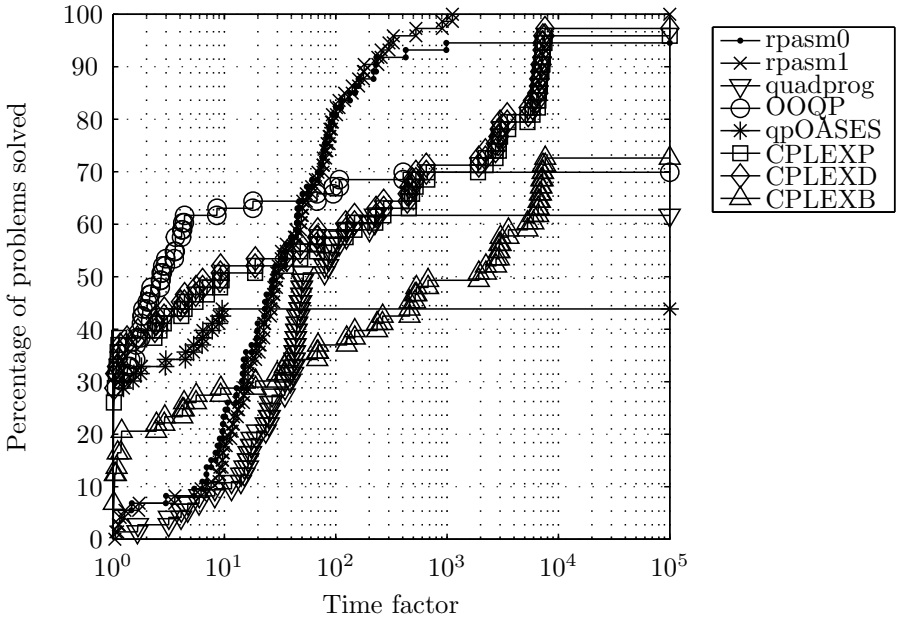

FIG. 6.1. *Performance comparison with loose residual threshold $\rho \leq$ 1e-2.*

There is a certain arbitrariness in the notion of a "solved problem" between the different codes. We choose to consider a problem as solved if ρ is less than or equal to a certain threshold. This approach is not unproblematic either: A not tight enough termination threshold of a code can lead to premature termination and the problem would be considered "not solved" by our criterion, although the method might have been able to recover a better solution with more iterations. This is especially an issue for Interior Point/Barrier methods. Thus the graphs in Figures 6.1 and 6.2 show reliability of the methods only in connection with their default settings. However, we are not aware of any simple procedure which would lead to a fairer comparison. Figure 6.1 shows the results with a relatively loose threshold of $\rho \leq$ 1e-2 and Figure 6.2 with a tighter threshold of $\rho \leq$ 1e-8.

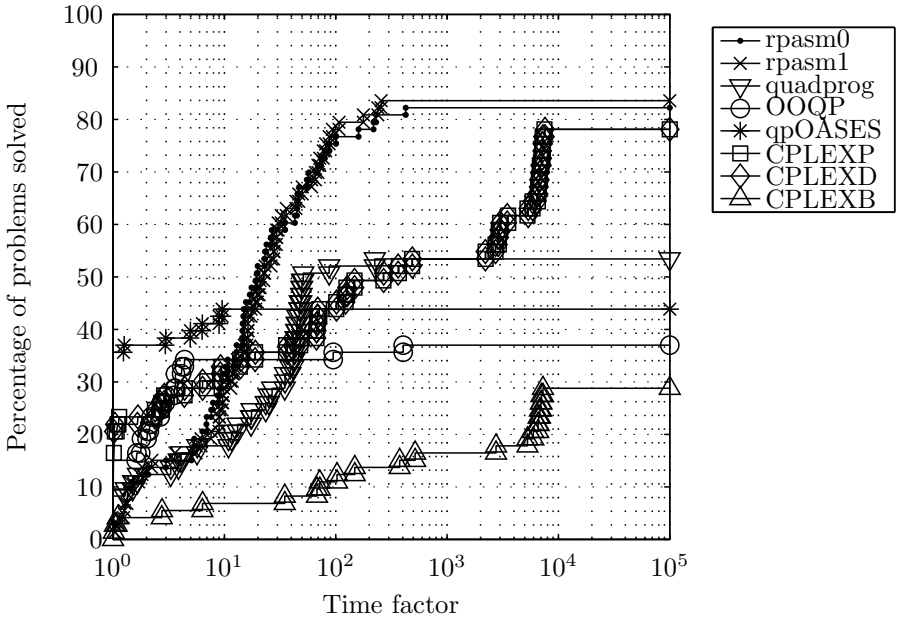

FIG. 6.2. *Performance comparison with tight residual threshold $\rho \leq$ 1e-8.*

6.2. Discussion of numerical results. We first discuss the results depicted in Figure 6.1 and continue with the differences to the tighter residual tolerance in Figure 6.2.

From Figure 6.1 we see that the newly developed code rpasm with iterative refinement is the only code which solves all of the problems to the prescribed accuracy. It actually solves all of them to a residual of $\rho \leq$ 5e-5, as can be seen from Table 8.1. The version of rpasm without iterative refinement fails on three problems (95 %). Furthermore, both versions of rpasm dominate quadprog both in runtime and the number of solved problems (62 %). The primal and dual versions of CPLEX are the second most reliable with 96 % and 97 %. As can be seen from Table 8.1, CPLEX solves no problem in less than 1.3 s, not even the small examples which are solved in a few miliseconds by rpasm. We suspect that this is due to a calling overhead in CPLEX, e.g., for license checking. This is also one reason why OOQP is much faster than the Barrier version of CPLEX, although they both solve roughly the same number of problems (70 % and 73 %, respectively). Although the code qpOASES is only appropriate for QPs with positive definite Hessian, which make up only 27 % of the considered problems, it still solves 44 % of the test problems. Additionally, we want to stress that those problems solved by qpOASES were indeed solved quickly.

Now we discuss the differences between Figure 6.2 and Figure 6.1, i.e., when switching to a tighter residual tolerance of $\rho \leq$ 1e-8: The ratio of solved problems drops dramatically for the Interior Point/Barrier methods (CPLEXB: 29 %, OOQP: 37 %). This is a known fact and the reason for the existence of crossover methods which refine the results of Interior Point/Barrier methods with an Active Set method. The code qpOASES still solves 44 % of the problems, which indicates that the solutions that qpOASES yields are of high quality. Furthermore, qpOASES is fast: It solves 36 % of the problems within 110 % of the time of the fastest method for each of these

problems. The number of problems solved by quadprog decreases to 53 %. The primal and dual Active Set versions of CPLEX solve 78 % of the problems. Only the code rpasm is able to solve more than 80 % of the problems to a residual of $\rho \leq$ 1e-8 (rpasm0: 82 %, rpasm1: 84 %).

We can conclude that the strategies proposed in §4 indeed lead to a reliable method for the solution of convex QPs.

7. Drawbacks of the proposed PASM. Although the method has proved to work successfully on the test set, the improvement in reliability is acheived only at the price of breaking the pure homotopy paradigm which complicates an otherwise straightforward proof of convergence for the method: Drift correction, ramping, and the flipping bounds strategy lead to jumps in the trajectories of $b(\tau), c^{\mathrm{l}}(\tau)$, and $c^{\mathrm{u}}(\tau)$ and thus to a sequence of (possibly nonphysical) homotopies. Proving the nonexistence or possibility of cycles through these strategies is beyond the scope of this paper.

One major area of application for the code qpOASES is online feedback control [9, 10]. Exploitation of the physical homotopy between two consecutive problems in time has proven to be of advantage in this case. The proposed strategies should thus be used with care in this application field.

8. Nonconvex Quadratic Programs. The flipping bounds strategy presented in §4.1 can also be extended to the case of nonconvex QPs with indefinite Hessian matrix B. When the Cholesky factorization or update breaks down due to a negative diagonal entry, we also flip instead of remove the constraint l. Hence the projected Hessian always stays positive definite. By the second order necessary optimality condition, the projected Hessian in every isolated local minimum of the nonconvex QP is guaranteed to be positive definite. Conversely, if strict complementarity holds at $\tau = 1$ we obtain a local minimum because the second order sufficient condition is satisfied. No guarantees can be given in the case of violation of strict complementarity.

Finding a global minimum of a nonconvex QP is known to be an NP-hard problem, even if the Hessian has only one single negative eigenvalue [28]. However, a local solution returned by the PASM can be refined by flipping all combinations of active bounds whose removal would lead to an indefinite projected Hessian and restarting the PASM for each of these flipped Active Sets, revealing again the combinatorial nature of finding the global solution of the nonconvex QP.

In the context of SQP with indefinite Hessian approximations (e.g., symmetric rank one updates, the exact Hessian, etc.), a local solution of a nonconvex QP is sufficient because the SQP method can only find local minima anyway.

For proof of concept we seek a local solution of the nonconvex problem

$$\min \ \frac{1}{2}\sum_{i=1}^{k-2}(x_{k+i+1} - x_{k+i})^2 - \frac{1}{2}\sum_{i=1}^{k=1}(x_{k-i} + x_{k+i} + \alpha_{k-i+1})^2 \tag{8.1a}$$

$$\text{s. t. } x_{k+i} - x_{i+1} + x_i = 0, \qquad i = 1, \ldots, k-1, \tag{8.1b}$$

$$\alpha_i \leq x_i \leq \alpha_{i+1}, \qquad i = 1, \ldots, k, \tag{8.1c}$$

$$0.4(\alpha_{i+2} - \alpha_i) \leq x_{k+i} \leq 0.6(\alpha_{i+2} - \alpha_i), \qquad i = 1, \ldots, k-1, \tag{8.1d}$$

with given constants $\alpha_i = 1 + 1.01^i, i = 1, \ldots, k-1$. We have adapted problem (8.1) from problem class 3 in Gould [16] by switching the sign in front of the second sum and the α terms in the objective. We start the computation with an initial guess of $x(0) = 0, y(0) = 0$, set the lower bounds of equations (8.1c) and (8.1d) active in the

initial working set, and adjust the variables via ramping (see §4.3). The changes of the working set are depicted in Figure 8.1 for $k = 100$ and therefore $n = 199, m = 298$. Row l of the depicted image corresponds to the working set in iteration l and column j corresponds to the status of constraint j in the working set when the iterations advance. The colors indicate constraints which are inactive (gray), active at the lower bound (black), or active at the upper bound (white). Thus direct transitions from black to white or vice versa along a vertical line indicate flipping bounds. We can observe that the chosen initial working set is considerably different to the final working set in the solution. Still the number of iterations is less than two times the number of constraints which indicates that the proposed method works efficiently on this instance of the nonconvex problem (8.1).

In the solution which corresponds to Figure 8.1, $n = 199$ out of m constraints are active and strict complementarity is satisfied. Thus we indeed have obtained a local optimum.

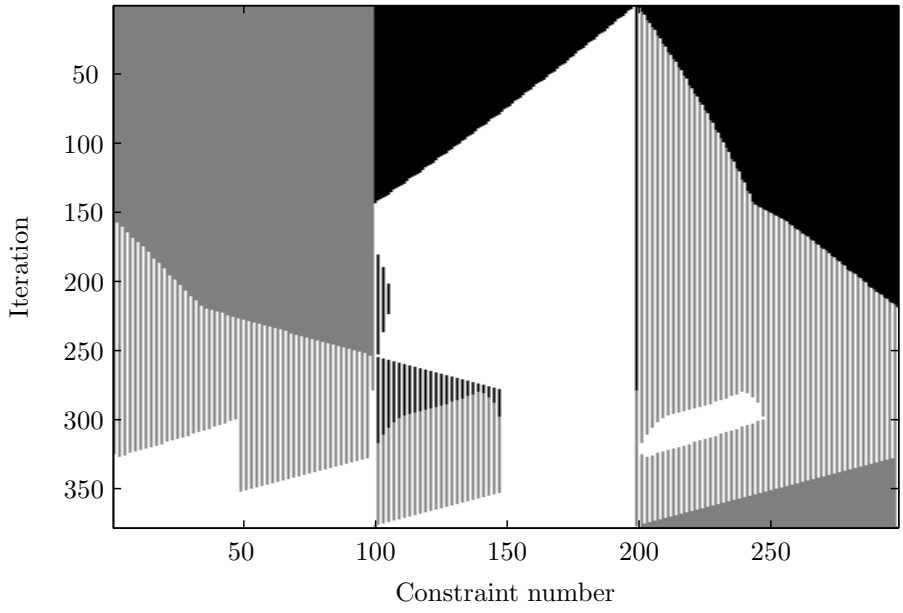

FIG. 8.1. *Active set changes for nonconvex problem* (8.1), $k = 100$. *Each line of the image corresponds to the working set in one iteration. The colors indicate constraints which are inactive (gray), active at the lower bound (black), or active at the upper bound (white). Direct transitions from black to white or vice versa along a vertical line indicate flipping bounds.*

problem			rpasm0		rpasm1		quadprog		OOQP		qpOASES		CPLEXP		CPLEXD		CPLEXB	
name	m	n	time	ρ	time	ρ	time	ρ	time	ρ	time	ρ	time	ρ	time	ρ	time	ρ
CVXQP1_M	500	1000	407.344	4e-10	529.085	2e-08	75.750	5e-10	3.197	6e-06		6e+00	1.757	2e-05	2.345	2e-06	2.096	3e-03
CVXQP1_S	50	100	0.576	5e-11	0.635	2e-11	0.114	2e-12	0.007	1e-08		6e+00	1.575	2e-11	1.600	2e-11	1.630	1e-06
CVXQP2_M	250	1000	221.006	5e-12	296.280	2e-11	331.569	2e-11	1.642	2e-06		6e+00	1.734	2e-07	1.834	2e-07	1.955	1e-04
CVXQP2_S	25	100	0.494	9e-13	0.540	2e-13	0.242	5e-13	0.006	4e-11	0.018	6e-13	1.663	5e-11	1.669	5e-11	1.632	2e-06
CVXQP3_M	750	1000	785.429	1e-09	968.948	2e-08	30.066	1e-08	5.125	4e-08		6e+00	1.839	5e-09	3.086	2e-09	2.576	3e-02
CVXQP3_S	75	100	0.684	5e-11	0.755	7e-11	0.051	1e-12	0.012	2e-10		6e+00	1.529	2e-12	1.530	5e-13	1.564	3e-06
DPKLO1	77	133	2.264	1e-14	2.472	1e-14	0.017	3e-14	0.010	4e-15	0.013	6e-14	1.523	3e-14	1.523	2e-14	1.531	3e-14
DUAL1	1	85	0.255	3e-15	0.298	2e-15	0.086	2e-15	0.003	2e-08	0.015	8e-15	1.526	2e-15	1.534	2e-15	1.550	1e-09
DUAL2	1	96	0.319	2e-15	0.350	1e-15	0.028	2e-15	0.003	2e-07	0.022	3e-15	1.566	2e-15	1.514	2e-15	1.581	2e-10
DUAL3	1	111	0.437	2e-15	0.469	3e-15	0.075	2e-15	0.003	4e-07	0.023	4e-15	1.532	3e-15	1.538	3e-15	1.554	1e-09
DUAL4	1	75	0.182	2e-15	0.196	2e-15	0.046	2e-15	0.002	4e-07	0.004	6e-15	1.562	1e-14	1.533	1e-14	1.584	4e-09
DUALC1	215	9	0.007	4e-12	0.008	6e-11	0.018	9e-10	0.036	2e-07	0.001	5e-12	1.531	1e-10	1.546	1e-10	1.551	2e-06
DUALC2	229	7	0.006	2e-12	0.007	2e-12	0.015	6e-11	0.042	3e-11	0.000	4e-12	1.546	5e-12	1.518	5e-12	1.552	7e-07
DUALC5	278	8	0.008	7e-13	0.008	3e-13	0.013	5e-13	0.056	7e-08	0.001	9e-13	1.544	2e-12	1.539	2e-12	1.556	4e-08
DUALC8	503	8	0.007	5e-11	0.009	1e-05	0.019	1e-10	0.282	7e-10	0.001	4e-11	1.551	1e-10	1.540	1e-10	1.560	7e-07
GENHS28	8	10	0.020	4e-16	0.021	4e-16	0.010	1e-15	0.000	1e-16	0.000	4e-16	1.528	2e-16	1.538	3e-15	1.566	2e-16
GOULDQP2	349	699	635.713	3e-09	721.674	3e-09	80.282	7e-14	0.644	2e-07		3e+01	1.696	5e-07	1.581	9e-07	1.571	2e-08
GOULDQP3	349	699	87.967	7e-15	110.301	7e-15	20.175	6e-14	1.427	4e-07	7.575	2e-13	1.581	1e-14	1.532	2e-14	1.523	2e-06
HS118	17	15	0.039	2e-15	0.041	3e-14	0.024	1e-13	0.002	3e-13	0.001	1e-13	1.524	2e-16	1.504	2e-16	1.528	3e-08
HS21	1	2	0.002	1e-16	0.002	2e-17	0.010	0e+00	0.001	2e-12	0.000	0e+00	1.512	0e+00	1.497	0e+00	1.519	6e-10
HS268	5	5	0.010	1e-11	0.010	1e-06	0.011	7e-12		0e+00	0.000	1e-11	1.526	2e-11	1.521	2e-11		2e+10
HS35	1	3	0.004	0e+00	0.004	9e-16	0.010	4e-16	0.001	2e-15	0.000	5e-15	1.535	0e+00	1.537	0e+00	1.529	2e-08
HS35MOD	1	3	0.005	5e-15	0.005	0e+00	0.010	0e+00	0.001	1e+00	0.000	0e+00	1.509	0e+00	1.522	0e+00	1.525	2e-08
HS51	3	5	0.006	5e-15	0.007	9e-16	0.010	9e-16	0.000	2e-32	0.000	4e-16	1.540	0e+00	1.504	0e+00	1.544	0e+00
HS52	3	5	0.010	2e-15	0.011	5e-16	0.009	2e-15	0.000	2e-16	0.000	2e-15	1.526	5e-15	1.547	8e-15	1.530	9e-16
HS53	3	5	0.010	9e-16	0.010	2e-15	0.015	2e-15	0.001	2e-08	0.000	9e-16	1.519	0e+00	1.542	0e+00	1.516	2e-10
HS76	3	4	0.004	2e-16	0.004	1e-16	0.011	2e-16	0.001	1e-13	0.000	5e-14	1.509	0e+00	1.508	0e+00	1.530	7e-09
KSIP	1001	20	1.718	2e-16	1.889	3e-16	0.248	2e-16	4.491	1e-08		1e+00	1.742	6e-16	1.587	1e-16	8.689	1e-09
LOTSCHD	7	12	0.023	9e-14	0.023	4e-13	0.013	1e-12	0.001	2e-07		1e+02	1.568	2e-13	1.591	2e-13	1.596	2e-06
MOSARQP2	600	900	359.303	4e-15	399.630	4e-15	697.550	2e+00	13.343	4e-07	13.942	3e-13	1.692	5e-14	1.555	7e-14	1.699	2e-06
PRIMAL1	85	325	14.951	1e-16	16.026	7e-17	2.232	4e-16	0.375	9e-08	0.573	6e-14	1.548	3e-14	1.513	3e-14	1.580	6e-10
PRIMAL2	96	649	156.441	2e-15	166.742	4e-16	18.018	4e-16	1.872	6e-07	9.832	2e-13	1.595	7e-15	1.552	7e-15	1.608	1e-10
PRIMAL3	111	745	255.530	4e-16	282.354	3e-16	27.584	1e-15	2.552	1e-06	15.190	3e-13	1.801	5e-14	1.587	5e-14	1.719	2e-09
PRIMALC1	9	230	0.430	3e-12	0.508	9e-13	1.877	1e-08	0.091	2e-11	0.022	1e-11	1.492	4e-12	1.480	2e-12	1.496	6e-08
PRIMALC2	7	231	0.209	2e-12	0.246	5e-13	1.976	2e-08	0.094	3e-10	0.021	8e-13	1.477	6e-12	1.483	6e-12	1.494	3e-08
PRIMALC5	8	287	0.285	2e-13	0.369	2e-13	3.603	2e-09	0.146	7e-10	0.041	2e-13	1.472	2e-13	1.516	1e-12	1.548	5e-08
PRIMALC8	8	520	2.113	1e-11	3.034	7e-12	26.585	4e-06	0.599	1e-11	0.333	4e-11	1.471	2e-10	1.464	3e-10	1.452	2e-05
QADLITTL	56	97	0.493	1e-09	0.517	8e-10	0.986	1e+02		9e+43		3e+03	1.414	7e-11	1.421	7e-11	1.433	9e-07
QAFIRO	27	32	0.026	1e-14	0.029	9e-16		1e+01	0.003	3e-09		4e+01	1.440	1e-14	1.448	1e-14	1.530	2e-10
QBANDM	305	472	71.053	2e-13	99.455	5e-12		4e+02		7e+12		7e+01	1.506	6e-13	1.515	6e-13	1.536	1e-06
QBEACONF	173	262	2.208	3e-11	2.594	4e-11		2e+03		8e+20		2e+03	1.511	3e-11	1.515	3e-11		5e+20
QBORE3D	233	315	4.148	9e-09	4.999	3e-09		3e+02		1e+40		3e+02	1.393	9e-13	1.392	5e-13	1.408	2e-06
QBRANDY	220	249	10.588	8e-11	13.225	1e-10		9e+02		0e+00		8e+01	1.386	4e-12	1.386	8e-12		7e+02
QCAPRI	271	353	27.599	1e-08	31.582	1e-08		4e+04		9e+19		3e+03	1.396	1e-08	1.397	1e-08		6e+20
QE226	223	282	14.447	5e-13	18.529	9e-14		2e+03		1e+20		3e+01	1.416	1e-13	1.420	1e-13		1e+20
QETAMACR	400	688	136.091	5e-05	181.163	5e-05		9e+02		8e+11		8e+02		9e+02	1.507	1e-07	1.565	2e-05
QFFFFF80	524	854		2e+05	1351.814	4e-06		6e+03		1e+25		2e+05	1.490	2e-10	1.502	2e-10		9e+21
QFORPLAN	161	421	49.313	6e-07	36.235	6e-07		4e+05		3e+10		7e+06	1.389	4e-07	1.397	5e-07	1.423	3e+01

continued on next page

continued from previous page

problem			rpasm0		rpasm1		quadprog		OOQP		qpOASES		CPLEXP		CPLEXD		CPLEXB	
name	m	n	time	ρ	time	ρ	time	ρ	time	ρ	time	ρ	time	ρ	time	ρ	time	ρ
QGROW15	300	645	58.214	3e-09	78.405	3e-09		7e+03	2.202	3e-08		7e+00	1.622	5e-07	1.630	3e-07	1.556	2e-02
QGROW22	440	946	191.875	1e-09	241.123	2e-09		7e+00	7.247	1e-06		7e+00	1.780	2e-08	1.960	2e-07	1.711	5e-03
QGROW7	140	301	7.046	1e-09	8.177	1e-09		7e+01	0.295	1e-09		7e+00	1.629	2e-08	1.637	2e-08	1.649	1e-03
QISRAEL	174	142	1.356	8e-06	1.570	3e-09		4e+04	0.163	5e-09		3e+03	1.537	1e-11	1.505	1e-11		2e+21
QPCBLEND	74	83	0.548	3e-15	0.581	2e-15	0.261	2e-14	0.056	2e-08		5e+00		5e+00		5e+00	1.666	4e-09
QPCBOEI1	351	384		2e+17	31.740	2e-09		5e+03	1.657	7e+10		3e+03	1.639	1e-07	1.576	3e-08	1.542	1e-03
QPCBOEI2	166	143	1.361	3e-06	1.505	2e-07		7e+03		2e+08		9e+03	1.485	2e-07	1.544	2e-07	1.619	3e-03
QPCSTAIR	356	467	35.642	1e-09	50.320	8e-10		5e+03		1e+31		3e+02	1.546	2e-08	1.559	1e-08	1.590	9e-05
QPTEST	2	2	0.003	2e-15	0.004	9e-16	0.010	2e-15	0.001	8e-13	0.000	3e-15	1.516	2e-15	1.523	2e-15	1.523	6e-09
QRECIPE	91	180	0.638	4e-14	0.729	8e-15		2e+00		2e+17		1e+01	1.517	2e-16	1.520	2e-16	1.535	3e+20
QSC205	205	203	1.494	3e-15	1.894	1e-16		3e+02		3e-04		1e+00	1.527	7e-18	1.528	7e-18		1e+18
QSCAGR25	471	500	75.680	5e-08	104.189	7e-09		6e+04	2.135	5e-08		7e+03	1.499	3e-09	1.504	3e-09	1.509	4e-04
QSCAGR7	129	140	2.098	6e-08	2.312	3e-09	0.249	5e-08	0.078	2e-10		7e+03	1.482	2e-11	1.486	2e-11	1.497	2e-04
QSCFXM1	330	457		2e+04	78.551	1e-06		4e+04		0e+00		1e+03	1.439	1e-07	1.450	1e-07		3e+20
QSCFXM2	660	914		3e+04	514.502	7e-07		7e+04		0e+00		1e+03	1.558	3e-07	1.571	3e-07		3e+20
QSCORPIO	388	358	14.351	2e-11	18.085	4e-11		2e+02		1e+52		2e+02	1.496	9e-13	1.508	9e-13		1e+17
QSCSD1	77	760	18.945	9e-09	28.110	9e-09		5e+00	2.388	4e-13		5e+00	1.471	4e-15	1.470	1e-09	1.487	1e-09
QSCTAP1	300	480	26.239	2e-11	39.236	1e-11		2e+02	2.406	2e-06		8e+01	1.490	5e-14	1.497	4e-14		8e+15
QSHARE1B	117	225	7.154	9e-08	7.137	1e-10	17.424	2e-07	0.162	7e-11		3e+03	1.488	1e-10	1.494	1e-10	1.510	2e-05
QSHARE2B	96	79	0.884	1e-08	0.936	1e-10	0.725	4e-10	0.031	3e-11		2e+01	1.480	1e-11	1.482	1e-11		9e+14
QSTAIR	356	467	34.228	1e-10	47.776	4e-10		4e+04	1.454	2e+09		3e+02	1.472	3e-09	1.489	3e-09		1e+17
S268	5	5	0.010	1e-11	0.011	1e-06	0.010	7e-12		0e+00	0.000	1e-11	1.428	2e-11	1.429	2e-11		2e+10
TAME	1	2	0.003	2e-16	0.003	0e+00	0.009	2e-16	0.000	1e-12	0.000	2e-16	1.436	0e+00	1.436	0e+00	1.436	0e+00
VALUES	1	202	1.183	1e-14	1.387	8e-15	3.850	1e-15	0.017	4e-09		1e+00		1e+00		1e+00		1e+00
ZECEVIC2	2	2	0.003	7e-16	0.003	0e+00	0.009	4e-16	0.001	4e-16		3e+00	1.425	0e+00	1.427	0e+00	1.439	3e-09

TABLE 8.1

Comparison of QP solvers on problems from the Maros-Mészáros test set [25] with at most $n = 1000$ *variables and* $m = 1001$ *two-sided inequality constraints (not counting variable bound constraints). The solution time in seconds and the optimality condition residual* ρ *are given for each solver. Failure of the solver is denoted by blank time fields.*

9. Conclusions and Outlook. We have identified numerical challenges in PASM, developed strategies to meet these challenges, and implemented a prototype Matlab® code named rpasm. The developed code has proved to solve all problems of the Maros-Mészáros test examples [25] with up to 1000 variables and up to 1001 two-sided constraints (plus potential simple variable bounds) with default parameter settings. In comparison, none of the other seven popular academic and commercial codes considered, covering Primal/Dual Active Set and Interior Point/Barrier methods, was able to solve all problems with default settings. We have found that the PASM implementation qpOASES delivers high-quality solutions very quickly. We conjecture that merging the fast linear algebra routines of qpOASES with the reliability improvements of rpasm will lead to a highly competitive general-purpose QP solver. Furthermore, we have proposed how the PASM can be used to find local solutions of nonconvex QPs. Work on a theoretical proof for convergence in the convex and nonconvex case is in progress.

Acknowledgements. The first author was supported by the German Research Foundation (DFG) within the priority program SPP1253 under grant BO864/12-1. Support by the German Federal Ministry of Education and Research (BMBF) under grant 03BONCHD is also gratefully acknowledged. The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement n° FP7-ICT-2009-4 248940. We thankfully acknowledge support by the Heidelberg Graduate School of Mathematical and Computational Methods for the Sciences funded by the Deutsche Forschungsgemeinschaft (DFG).

REFERENCES

[1] *The HSL mathematical software library.* http://www.hsl.rl.ac.uk/, 2007.
[2] *The MOSEK optimization tools manual, version 6.0 (revision 85).* http://www.mosek.com/, 2010.
[3] M.J. Best, *An Algorithm for the Solution of the Parametric Quadratic Programming Problem*, Applied Mathematics and Parallel Computing, Physica-Verlag, Heidelberg, 1996, ch. 3, pp. 57–76.
[4] J. Blue, P. Fox, W. Fullerton, D. Gay, E. Grosse, A. Hall, L. Kaufman, W. Petersen, and N. Schryer, *PORT mathematical subroutine library.* http://www.bell-labs.com/project/PORT/, 1997.
[5] J. Dahl and L. Vandenberghe, *CVXOPT user's guide, release 1.1.3.* http://abel.ee.ucla.edu/cvxopt/userguide/index.html, September 2010.
[6] G.B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, 1963.
[7] L. Di Gaspero, *QuadProg++.* http://www.diegm.uniud.it/digaspero/index.php/software/, 2010.
[8] Fair Isaac Corporation, *FICO(TM) Xpress Optimization Suite, Xpress-Optimizer Reference manual, Release 20.00*, Warwickshire, UK, 2009.
[9] H.J. Ferreau, *qpOASES – an open-source implementation of the online active set strategy for fast model predictive control*, in Proceedings of the Workshop on Nonlinear Model Based Control – Software and Applications, Loughborough, 2007.
[10] H.J. Ferreau, H.G. Bock, and M. Diehl, *An online active set strategy to overcome the limitations of explicit MPC*, International Journal of Robust and Nonlinear Control, 18 (2008), pp. 816–830.
[11] R. Fletcher, *Resolving degeneracy in quadratic programming*, Annals of Operations Research, 46-47 (1993), pp. 307–334.
[12] E.M. Gertz and S.J. Wright, *Object-oriented software for quadratic programming*, ACM Transactions on Mathematical Software, 29 (2003), pp. 58–81.
[13] P.E. Gill, W. Murray, and M.A. Saunders, *User's Guide For QPOPT 1.0: A Fortran Package For Quadratic Programming*, 1995.

[14] P.E. Gill, W. Murray, M.A. Saunders, and M.H. Wright, *Procedures for optimization problems with a mixture of bounds and general linear constraints*, ACM Transactions on Mathematical Software, 10 (1984), pp. 282–298.
[15] J. Gondzio, *HOPDM (version 2.12) – a fast LP solver based on a primal-dual interior point method*, European Journal of Operational Research, 85 (1995), pp. 221–225.
[16] N.I.M. Gould, *An algorithm for large-scale quadratic programming*, IMA Journal of Numerical Analysis, 11 (1991), pp. 299–324.
[17] N.I.M. Gould, D. Orban, and P.L. Toint, *CUTEr testing environment for optimization and linear algebra solvers*. http://cuter.rl.ac.uk/cuter-www/.
[18] N.I.M. Gould, D. Orban, and Ph.L. Toint, *GALAHAD, a library of thread-safe Fortran 90 packages for large-scale nonlinear optimization*, ACM Trans. Math. Software, 29 (2004), pp. 353–372.
[19] N.I.M. Gould and P.L. Toint, *A quadratic programming bibliography*, Tech. Report 2000-1, Rutherford Appleton Laboratory, Computational Science and Engineering Department, March 2010.
[20] IBM Corp., *IBM ILOG CPLEX V12.1, User's Manual for CPLEX*, New York, USA, 2009.
[21] C. Kirches, H.G. Bock, J.P. Schlöder, and S. Sager, *Block structured quadratic programming for the direct multiple shooting method for optimal control*, Optimization Methods and Software, (2010). (available online).
[22] C. Kirches, H.G. Bock, J.P. Schlöder, and S. Sager, *A factorization with update procedures for a KKT matrix arising in direct optimal control*, Mathematical Programming Computation, (2010). (submitted). Available Online: http://www.optimization-online.org/DB_HTML/2009/11/2456.html.
[23] E. Kostina, *The long step rule in the bounded-variable dual simplex method: Numerical experiments*, Mathematical Methods of Operations Research, 55 (2002), pp. 413–429.
[24] S. Lauer, *SQP–Methoden zur Behandlung von Problemen mit indefiniter reduzierter Hesse–Matrix*, Diploma thesis, Ruprecht–Karls–Universität Heidelberg, February 2010.
[25] I. Maros and C. Mészáros, *A repository of convex quadratic programming problems*, Optimization Methods and Software, 11 (1999), pp. 671–681.
[26] The Mathworks, *Matlab optimization toolbox user's guide*. http://www.mathworks.com/products/optimization/, 2010.
[27] C. Mészáros, *The BPMPD interior point solver for convex quadratic problems*, Optimization Methods and Software, 11 (1999), pp. 431–449.
[28] K.G. Murty, *Some NP-complete problems in quadratic and nonlinear programming*, Mathematical Programming, 39 (1987), pp. 117–129.
[29] J. Nocedal and S.J. Wright, *Numerical Optimization*, Springer Verlag, Berlin Heidelberg New York, 2nd ed., 2006. ISBN 0-387-30303-0 (hardcover).
[30] S. Sager, *Lange Schritte im Dualen Simplex-Algorithmus*, Diploma thesis, Universität Heidelberg, 2001.
[31] B.A. Turlach, *QuadProg (quadratic programming routines), release 1.4*. http://school.maths.uwa.edu.au/~berwin/software/quadprog.html, July 1998.
[32] R.J. Vanderbei, *LOQO: An interior point code for quadratic programming*, Optimization Methods and Software, 11 (1999), pp. 451–484.
[33] X. Wang, *Resolution of ties in parametric quadratic programming*, master's thesis, University of Waterloo, Ontario, Canada, 2004.
[34] J.H. Wilkinson, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.
[35] X. Zhang and Y. Ye, *User's guide of COPL_QP, computational optimization program library: Convex quadratic programming*. http://www.stanford.edu/~yyye/Col.html, July 1998.