Mathematical Programming Computation (2019) 11:421–455
https://doi.org/10.1007/s12532-019-00154-6

FULL LENGTH PAPER

Solving quadratic programs to high precision using scaled iterative refinement

Tobias Weber[1] · **Sebastian Sager**[1] · **Ambros Gleixner**[2]

Received: 20 November 2017 / Accepted: 22 December 2018 / Published online: 6 February 2019

Abstract
Quadratic optimization problems (QPs) are ubiquitous, and solution algorithms have matured to a reliable technology. However, the precision of solutions is usually limited due to the underlying floating-point operations. This may cause inconveniences when solutions are used for rigorous reasoning. We contribute on three levels to overcome this issue. First, we present a novel refinement algorithm to solve QPs to arbitrary precision. It iteratively solves refined QPs, assuming a floating-point QP solver oracle. We prove linear convergence of residuals and primal errors. Second, we provide an efficient implementation, based on `SoPlex` and `qpOASES` that is publicly available in source code. Third, we give precise reference solutions for the Maros and Mészáros benchmark library.

Keywords Quadratic programming · Iterative refinement · Active set · Rational calculations

Mathematics Subject Classification 90C20 · 90-08 · 90C55

The software that was reviewed as part of this submission was given the DOI (Digital Object Identifier) https://doi.org/10.5281/zenodo.2532184.

This project has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (Grant Agreement No 647573), from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)—314838170, GRK 2297 MathCoRe, and from the German Federal Ministry of Education and Research as part of the *Research Campus MODAL* (BMBF Grant Number 05M14ZAM), all of which is gratefully acknowledged.

✉ Tobias Weber
Tobias.Weber@ovgu.de

Ambros Gleixner
gleixner@zib.de

[1] Institute of Mathematical Optimization, Otto von Guericke University, Universitätsplatz 2, 02-204, 39106 Magdeburg, Germany

[2] Department of Mathematical Optimization, Zuse Institute Berlin, Takustr. 7, 14195 Berlin, Germany

1 Introduction

Quadratic optimization problems (QPs) are optimization problems with a quadratic objective function and linear constraints. They are of interest directly, e.g., in portfolio optimization or support vector machines [1]. They also occur as subproblems in sequential quadratic programming, mixed-integer quadratic programming, and nonlinear model predictive control. Efficient algorithms are usually of active set, interior point, or parametric type. Examples of QP solvers are `BQPD` [5], `CPLEX` [2], `Gurobi` [15], `qp_solve` [12], `qpOASES` [4], and `QPOPT` [7]. These QP solvers have matured to reliable tools and can solve convex problems with many thousands, sometimes millions of variables. However, they calculate and check the solution of a QP in floating-point arithmetic. Thus, the claimed precision may be violated and in extreme cases optimal solutions might not be found. This may cause inconveniences, especially when solutions are used for rigorous reasoning.

One possible approach is the application of interval arithmetic. It allows to include uncertainties as lower and upper bounds on the modeling level, see [16] for a survey for the case of linear optimization. As a drawback, all internal calculations have to be performed with interval arithmetic. Standard solvers can not be used any more and their conversion to interval arithmetic becomes non-trivial when division by intervals containing zero is encountered. In any case, computation times increase and solutions may be very conservative.

We are aware of only one advanced algorithm that solves QPs exactly over the rational numbers. It is designed to tackle problems from computational geometry with a small number of constraints or variables [6]. Based on the classical QP simplex method [23], it replaces critical calculations inside the QP solver by their rational counterparts. Heuristic decisions that do not affect the correctness of the algorithm are performed in fast floating-point arithmetic.

In this paper we propose a novel algorithm that can use efficient floating-point QP solvers as a black box. Our method is inspired by iterative refinement, a standard procedure to improve the accuracy of an approximate solution for a system of linear equalities [22]: The residual of the approximate solution is calculated, the linear system is solved again with the residual as a right-hand side, and the new solution is used to refine the old solution, thus improving its accuracy. A generalization of this idea to the solution of optimization problems needs to address several difficulties: most importantly, the presence of inequality constraints; the handling of optimality conditions; and the numerical accuracy of floating-point solvers in practice.

For linear programs (LPs) this has first been developed in [9,10]. The approach refines primal-dual solutions of the Karush–Kuhn–Tucker (KKT) conditions and utilizes scaling and calculations in rational arithmetic. We generalize this method further and discuss the specific issues due to the presence of a quadratic objective function. The fact that the approach carries over from LP to QP was remarked in [8]. Here we provide the details, provide a general lemma showing how the residuals bound the primal and dual iterates, and analyze the computational behavior of the algorithm based on an efficient implementation that is made publicly available in source code and can be used freely for research purposes.

The idea to refine QP solutions has been explored before. In [18] a reference QP is solved in floating-point precision by an active-set method and its basis matrix factorization is used to set up and solve a transformed QP. The error introduced by this matrix approximation is then corrected by an iterative refinement procedure. This leads to a sequence of QPs with differing right-hand side vectors that compute refinement steps converging to a solution of the original QP in floating-point precision. The main purpose here is to speed up the QP solution process by avoiding to factorize the basis matrix of the QP.

Other methods use iterative refinement to deal with errors introduced by the implicit treatment of the constraints. In [20] one part of the constraints of a nonlinear programming problem is treated implicitly inside a sequential quadratic programming scheme. The nullspace of these equations is only approximated and the refinement equation for the error is included into the QPs to stabilize the inexact newton scheme. In [13] a QP is solved by a conjugate gradient algorithm implicitly treating all constraints. In order to cope with numerical inaccuracies in the projection of the iterates onto the feasible set, different iterative refinement extensions to the algorithm are proposed. These approaches use iterative refinement on a lower level inside of a dedicated algorithm. In contrast, the iterative refinement scheme is designed to employ any floating-point QP algorithm as a black-box subroutine.

The paper is organized as follows. In Sect. 2 we define and discuss QPs and their refined and scaled counterparts. We give one illustrating and motivating example for scaling and refinement. In Sect. 3 we formulate an algorithm and prove its convergence properties. In Sect. 4 we consider performance issues and describe how our implementation based on `SoPlex` and `qpOASES` can be used to calculate solutions for QPs with arbitrary precision. In Sect. 5 we discuss run times and provide solutions for the Maros and Mészáros benchmark library [19]. We conclude in Sect. 6 with a discussion of the results and give directions for future research and applications of the algorithm.

In the following we will use $\|\cdot\|$ for the maximum norm $\|\cdot\|_\infty$. The maximal entry of a vector $\max_i\{v_i\}$ is written as $\max\{v\}$. Inequalities $a \leq b$ for $a, b \in \mathbb{Q}^n$ hold componentwise. $\mathbb{Q}_+$ denotes the set of positive rationals.

2 Refinement and scaling of quadratic programs

In this section we collect some basic definitions and results that will be of use later on. We consider convex optimization problems of the following form.

Definition 1 (*Convex QP with Rational Data*) Let a symmetric matrix $Q \in \mathbb{Q}^{n\times n}$, a matrix $A \in \mathbb{Q}^{m\times n}$, and vectors $c \in \mathbb{Q}^n, b \in \mathbb{Q}^m, l \in \mathbb{Q}^n$ be given. We consider the quadratic optimization problem (QP)

$$\begin{aligned} \min_x \ & \tfrac{1}{2} x^T Q x + c^T x \\ \text{s.t. } & Ax = b \\ & x \geq l \end{aligned} \tag{P}$$

assuming that (P) is feasible and bounded, and Q is positive semi-definite on the feasible set.

A point $x^* \in \mathbb{Q}^n$ is a global optimum of (P) if and only if it satisfies the Karush–Kuhn–Tucker (KKT) conditions [3], i.e., if multipliers $y^* \in \mathbb{Q}^m$ exist such that

$$Ax^* = b \tag{1a}$$
$$x^* \geq l \tag{1b}$$
$$A^T y^* \leq Qx^* + c \tag{1c}$$
$$(Qx^* + c - A^T y^*)^T (x^* - l) = 0. \tag{1d}$$

The pair (x^*, y^*) is then called KKT pair of (P). Primal feasibility is given by (1a) and (1b), dual feasibility by (1c), and complementary slackness by (1d). Refinement of this system of linear (in-)equalities is equivalent to the refinement of (P).

Definition 2 (*Refined QP*) Let the QP (P), scaling factors $\Delta_P, \Delta_D \in \mathbb{Q}_+$ and vectors $x^* \in \mathbb{Q}^n$, $y^* \in \mathbb{Q}^m$ be given. We define the refined QP as

$$\begin{aligned} \min_x \ & \frac{1}{2} x^T \frac{\Delta_D}{\Delta_P} Q x + (\Delta_D \hat{c})^T x \\ \text{s.t. } & Ax = \Delta_P \hat{b} \\ & x \geq \Delta_P \hat{l}, \end{aligned} \tag{P^Δ}$$

where $\hat{c} = Qx^* + c - A^T y^*$, $\hat{b} = b - Ax^*$, and $\hat{l} = l - x^*$.

The following theorem is the basis for our theoretical and algorithmic work. It is a generalization of iterative refinement for LP and was formulated and proven in [8, Theorem 5.2]. Again, primal feasibility refers to (1a) and (1b), dual feasibility to (1c), and complementary slackness to (1d).

Theorem 3 (QP Refinement) *Let the QP (P), scaling factors $\Delta_P, \Delta_D \in \mathbb{Q}_+$, vectors $x^* \in \mathbb{Q}^n$, $y^* \in \mathbb{Q}^m$, and the refined QP (P^Δ) be given. Then for any $\hat{x} \in \mathbb{R}^n$, $\hat{y} \in \mathbb{R}^m$ and tolerances $\epsilon_P, \epsilon_D, \epsilon_S \geq 0$:*

1. *$\hat{x}$ is primal feasible for (P^Δ) within an absolute tolerance ϵ_P if and only if $x^* + \frac{\hat{x}}{\Delta_P}$ is primal feasible for (P) within ϵ_P / Δ_P.*
2. *$\hat{y}$ is dual feasible for (P^Δ) within an absolute tolerance ϵ_D if and only if $y^* + \frac{\hat{y}}{\Delta_D}$ is dual feasible for (P) within ϵ_D / Δ_D.*
3. *$\hat{x}$, $\hat{y}$ satisfy complementary slackness for (P^Δ) within an absolute tolerance ϵ_S if and only if $y^* + \frac{\hat{y}}{\Delta_D}$, $x^* + \frac{\hat{x}}{\Delta_P}$ satisfy complementary slackness for (P) within $\epsilon_S / (\Delta_P \Delta_D)$.*

For illustration, consider the following example.

Example 4 **(QP Refinement)** Consider the QP with two variables

$$\begin{aligned} \min_x \ & \frac{1}{2}(x_1^2 + x_2^2) + x_1 + (1 + 10^{-6}) x_2 \\ \text{s.t. } & x_1 + x_2 = 10^{-6} \\ & x_1, x_2 \geq 0. \end{aligned}$$

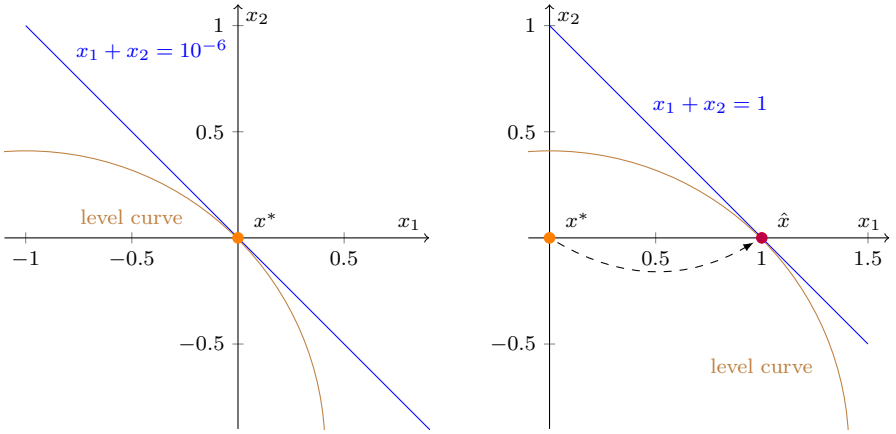

Fig. 1 Illustration of nominal QP (left) and of refined QP (right) for Example 4. The scaled (and shifted) QP (P^{Δ}) acts as a shift and zoom for (P), allowing to correct the solution x^* (orange dot) from $(0, 0)$ to $(10^{-6}, 0)$

An approximate solution to a tolerance of 10^{-6} is $x_1^* = x_2^* = 0$ with dual multiplier $y^* = 1$. This solution is slightly primal and dual infeasible, but the solver can not recognize this on this scale. The situation is depicted in Fig. 1 on the left.

The point x^* seems to be the optimal solution satisfying the equality constraint and the brown circle representing the level curve of the objective function indicates the optimality. The corresponding violations are $\hat{l} = (0, 0)^T$, $\hat{b} = 10^{-6}$, and $\hat{c} = (0, 10^{-6})^T$. The refined QP is

$$\begin{aligned} \min_x \ & \frac{1}{2}(x_1^2 + x_2^2) + x_2 \\ \text{s.t. } & x_1 + x_2 = 1 \\ & x_1, x_2 \geq 0 \end{aligned}$$

with scaling factors $\Delta_P = \Delta_D = 10^6$. The optimal solution to this problem is $\hat{x}_1 = 1, \hat{x}_2 = 0$ with multiplier $\hat{y} = 1$. This situation is depicted in Fig. 1 on the right. The initial point x^* is obviously not the optimal solution and the solution to the refined problem is $\hat{x}$. The refined solution is $x^* + \hat{x}/\Delta_P = (10^{-6}, 0)^T$ and $y^* + \hat{y}/\Delta_D = 1 + 10^{-6}$. These values are primal and dual feasible in the original problem.

Note that we restrict the presentation to QPs with rational data because this parallels the implementation on which the computational study in Sect. 5 is based. However, the method is applicable to QPs over any field that is dense in $\mathbb{R}$ and contains the floating-point numbers returned by the QP solver. In practice, it is merely required that we can perform exact arithmetic over the field and that numbers from the field can be rounded to a floating-point approximation with small error. This certainly holds for $\mathbb{Q}$, for which software support is provided by several libraries.

The increased arithmetic cost of rational arithmetic stems from storing arbitrarily large integer numerators and denominators and reducing them to be co-prime by regularly computing their greatest common divisor. Floating-point numbers such as

standard double-precision numbers specified by the IEEE Microcomputer Standards Committee [17] are rational numbers usually stored in binary representation

$$\pm 1.q_1q_2q_3\ldots q_{52}\cdot 2^e$$

with $q_i \in \{0, 1\}$ and $e \in \{-1022, -1021, \ldots, 1023\}$, plus special bit patterns for zero, plus and minus infinity, and the result of undefined arithmetic operations ("not-a-number"). This gives 15 to 17 significant decimal digits to which each rational number can be rounded. Specifically, we will round the rational data of the QPs P and (P^{Δ}) defined above in order to pass their floating-point versions $\tilde{P}$ and $\tilde{P}^{\Delta}$, respectively, to the underlying QP solver. Conversely, each floating-point number is a rational number, so no precision is lost when using solutions from the QP solver for a correction step as in Theorem 3. For further details on rational and floating-point arithmetic we refer to [11,14].

3 The iterative refinement algorithm for quadratic programming

To solve quadratic programs to arbitrary, *a priori* specified precision, we apply the refinement idea from the previous section iteratively as detailed in Algorithm 1. Algorithm 1 expects QP data (Q, A, c, b, l) in rational precision, primal and dual termination tolerances (ϵ_P, ϵ_D), complementary slack termination tolerance (ϵ_S), scaling limit $\alpha > 1$ and iteration limit k_{max}. First the rounded QP $(\tilde{P})$ is solved with a floating-point QP solver oracle which returns optimal primal and dual solution vectors (*Line 2*). In *Line 3* the main loop begins. The primal violations for constraints ($\hat{b}$, *Line 4*) and for bounds ($\hat{l}$, *Line 5*) are calculated. The maximal primal violation is saved as δ_P in *Line 6*. The reduced cost vector $\hat{c}$ and its maximal violation δ_D are calculated in *Lines 7–8*. In *Line 9* the scaling factor Δ_k is chosen as the maximum of $\alpha\Delta_{k-1}$ and the inverses of the violations δ_P and δ_D. The complementary slack violation δ_S is calculated in *Line 10*. If the primal, dual and complementary slack violations are already below the specified tolerances the loop is stopped (*Lines 11–12*) and the optimal solution is returned (*Line 17*). Else (*Line 13*) the refined, scaled, and rounded QP $(\tilde{P}^{\Delta_k})$ is solved with the floating-point QP oracle in *Line 14*. We save the floating-point optimal primal and dual solution vectors (*Line 15*). We scale and add them to the current iterate (x_k, y_k) to obtain (x_{k+1}, y_{k+1}), *Line 16*.

Note that all calculations except the expensive solves of the QPs are done in rational precision. Algorithm 1 uses only one scaling factor Δ_k for primal and dual infeasibility to avoid the scaling of the quadratic term of the objective. Keeping this matrix and the constraint matrix A fixed gives QP solvers the possibility to reuse the internal factorization of the basis system between refinements, as the transformation does not change the basis. Hence one can perform hotstarts with the underlying QP solver which is crucial for the practical performance of the algorithm. This comes at the cost of only scaling either primal or dual infeasibilities as required, especially if they differ a lot, possibly slowing convergence.

To investigate the performance of the algorithm we make, in analogy with the LP case [10, Ass. 1], the following assumption.

Algorithm 1 Iterative QP Refinement (IQPR)

Input: (P) in rational precision, termination tolerances ϵ_P and ϵ_D and ϵ_S, scaling limit $\alpha > 1$, iteration limit k_{max}
Initialization: $\Delta_0 \leftarrow 1$, solve ($\tilde{P}$) approximately, save optimal (x_1, y_1)
for $k \leftarrow 1, 2, ..., k_{max}$ **do**
  $\hat{b} \leftarrow b - Ax_k$
  $\hat{l} \leftarrow l - x_k$
  $\delta_P \leftarrow \max\left\{0, \|\hat{b}\|, \max\{\hat{l}\}\right\}$
  $\hat{c} \leftarrow Qx_k + c - A^T y_k$
  $\delta_D \leftarrow \max\left\{0, \max\{-\hat{c}\}\right\}$
  $\Delta_k \leftarrow \min\left\{\delta_P^{-1}, \delta_D^{-1}, \alpha\Delta_{k-1}\right\}$
  $\delta_S \leftarrow \sum_i \hat{l}_i\hat{c}_i$
  if $\delta_P \le \epsilon_P$ and $\delta_D \le \epsilon_D$ and $\delta_S \le \epsilon_S$ **then**
    break
  else
    solve $(\tilde{P}^{\Delta_k})$ approximately
    $(x^*, y^*) \leftarrow$ KKT pair returned as optimal
    $(x_{k+1}, y_{k+1}) \leftarrow (x_k, y_k) + \frac{(x^*, y^*)}{\Delta_k}$
Return: (x_k, y_k)

Assumption 5 (*QP Solver Accuracy*) We assume that there exists $\epsilon \in [0, 1)$ and $\sigma \ge 0$ such that the QP solver oracle returns for all rounded QPs $(\tilde{P}^{\Delta_k})$ solutions $(\bar{x}, \bar{y})$ that satisfy

$$\begin{aligned} \|A\bar{x} - \Delta_k\hat{b}\| &\le \epsilon \\ \bar{x} &\ge \Delta_k\hat{l} - \mathbb{1}\epsilon \\ Q\bar{x} + \Delta_k\hat{c} &\ge A^T\bar{y} - \mathbb{1}\epsilon \\ |(Q\bar{x} + \Delta_k\hat{c} - A^T\bar{y})^T(\bar{x} - \Delta_k\hat{l})| &\le \sigma \end{aligned}$$

with respect to the rational input data of QPs (P^{Δ_k}).

Note that this ϵ corresponds to a termination tolerance passed to a floating-point solver, while the algorithm uses overall termination tolerances ϵ_P and ϵ_D and a scaling limit $\alpha > 1$ per iteration. We denote $\tilde{\epsilon} = \max\{1/\alpha, \epsilon\}$.

Lemma 6 (Termination and Residual Convergence) *Algorithm 1 applied to a primal and dual feasible QP (P) and using a QP solver that satisfies Assumption 5 will terminate in at most*

$$k_{max} = \max\{\log(\epsilon_P)/\log(\tilde{\epsilon}),\ \log(\epsilon_D)/\log(\tilde{\epsilon}),\ \log(\epsilon_S/\sigma)/(2\log(\tilde{\epsilon})) + 1\} \quad (2)$$

iterations. Furthermore, after each iteration $k = 1, 2, \ldots$ the primal-dual iterate (x_k, y_k) and the scaling factor Δ_k satisfy

$$\Delta_k \ge 1/\tilde{\epsilon}^k \quad (3a)$$

$$\|Ax_k - b\| \leq \tilde{\epsilon}^k \tag{3b}$$
$$x_k - l \geq -\mathbb{1}\tilde{\epsilon}^k \tag{3c}$$
$$Qx_k + c - A^T y_k \geq -\mathbb{1}\tilde{\epsilon}^k \tag{3d}$$
$$|(Qx_k + c - A^T y_k)^T (x_k - l)| \leq \sigma\tilde{\epsilon}^{2(k-1)}. \tag{3e}$$

Proof We prove (3) by induction over k, starting with $k = 1$. As $\tilde{\epsilon} \geq \epsilon$, the claims (3b–3e) follow directly from Assumption 5. Using *Lines 6*, *4–5*, and Assumption 5 we obtain

$$\delta_P = \max\left\{0, \|\hat{b}\|, \max\{\hat{l}\}\right\} = \max\{0, \|Ax_1 - b\|, \max\{l - x_1\}\} \leq \epsilon$$

and with *Lines 8,7* and Assumption 5

$$\delta_D = \max\left\{0, \max\{-\hat{c}\}\right\} = \max\left\{0, \max\{Qx_1 + c - A^T y_1\}\right\} \leq \epsilon.$$

Thus from *Line 9* we have

$$\Delta_1 = \min\left\{\delta_P^{-1}, \delta_D^{-1}, \alpha\Delta_0\right\} \geq \min\left\{\epsilon^{-1}, \epsilon^{-1}, \alpha\right\} \geq \tilde{\epsilon}^{-1}$$

and hence claim (3a) for the first iteration.

Assuming (3) holds for k we know that $\delta_{P,k}, \delta_{D,k} \leq \tilde{\epsilon}^k$ and $\Delta_k \geq 1/\tilde{\epsilon}^k$. With the scaling factor Δ_k using $x^* = x_k$ and $y^* = y_k$ we scale the QP (P) as in Theorem 3 and hand it to the QP solver. By Theorem 3 this scaled QP is still primal and dual feasible and by Assumption 5 the solver hands back a solution $(\hat{x}, \hat{y})$ with tolerance $\epsilon \leq \tilde{\epsilon}$. Therefore using Theorem 3 again the next refined iterate (x_{k+1}, y_{k+1}) has a tolerance in QP (P) of $\tilde{\epsilon}/\Delta_k \leq \tilde{\epsilon}^{k+1}$, which proves (3b–3d).

With the same argument the solution $(\hat{x}, \hat{y})$ violates complementary slackness by σ in the scaled QP (Assumption 5) and the refined iterate (x_{k+1}, y_{k+1}) violates complementary slackness in QP (P) by $\sigma/\Delta_k^2 \leq \sigma\tilde{\epsilon}^{2k}$ proving (3e).

We have now $\delta_{P,k+1}, \delta_{D,k+1} \leq \tilde{\epsilon}^{k+1}$. Also it holds that $\alpha\Delta_k \geq \alpha/\tilde{\epsilon}^k \geq 1/\tilde{\epsilon}^{k+1}$. *Line 9* of Algorithm 1 gives

$$\Delta_{k+1} \geq 1/\tilde{\epsilon}^{k+1},$$

proving (3a).

Then (2) follows by assuming the slowest convergence rate of the primal, dual and complementary violations and by comparing this with the termination condition in *Line 11* of Algorithm 1

$$\tilde{\epsilon}^k \leq \epsilon_P,\ \tilde{\epsilon}^k \leq \epsilon_D, \sigma\tilde{\epsilon}^{2(k-1)} \leq \epsilon_S.$$

This is equivalent to (2). □

The results show that even though we did not use the violation of the complementary slackness to choose the scaling factor in Algorithm 1, the complementary slackness violation is bounded by the square of $\tilde{\epsilon}$.

Remark 7 **(Nonconvex QPs)** Algorithm 1 can also be used to calculate high precision KKT pairs of nonconvex QPs. If the black box QP solver hands back local solutions of the quality specified in Assumption 5 Lemma 6 holds as well for nonconvex QPs then Algorithm 1 returns a high precision local solution.

However, assuming strict convexity, an even stronger result holds. Inspired by the result for the equality-constrained QP [3, Proposition 2.12] we investigate how this right-hand side convergence of the KKT conditions is related to the primal-dual solution.

Lemma 8 (Primal and Dual Solution Accuracy) *Let QP (P) be given and be strictly convex, the minimal and maximal eigenvalues of Q be $\lambda_{\min}(Q)$ and $\lambda_{\max}(Q)$, respectively, and the minimal nonzero singular value of A be $\sigma_{\min}(A)$. Let the KKT conditions (1) hold for (x^*, y^*, z^*), i.e.,*

$$Ax^* = b \tag{4a}$$
$$A^T y^* + z^* = Qx^* + c \tag{4b}$$
$$z^{*T}(x^* - l) = 0 \tag{4c}$$
$$x^* \geq l \tag{4d}$$
$$z^* \geq 0 \tag{4e}$$

and the perturbed KKT conditions for perturbations $e \in \mathbb{Q}^m$, $g, f, h \in \mathbb{Q}^n$, and $i \in \mathbb{Q}$ hold for (x, y, z), i.e.,

$$Ax = b + e \tag{5a}$$
$$A^T y + z = Qx + c + g \tag{5b}$$
$$z^T(x - l) = i \tag{5c}$$
$$x \geq l + f \tag{5d}$$
$$z \geq h. \tag{5e}$$

Denote

$$a := \frac{\lambda_{\max}(Q)\|e\|_2}{2\sigma_{\min}(A)} + \lambda_{\max}(Q)\lambda_{\min}(Q)\|g\|_2/2$$
$$d := \lambda_{\max}(Q)\|i - h^T(x^* - l) - z^{*T} f\|_2.$$

Then

$$\|A^T(y - y^*) + (z - z^*)\|_2 \leq a + \sqrt{a^2 + d} \tag{6}$$

and

$$\|x - x^*\|_2 \leq \lambda_{\min}(Q)(a + \sqrt{a^2 + d}) + \lambda_{\min}(Q)\|g\|_2. \tag{7}$$

Proof By (4a) and (5a) we have that $A(x - x^*) = e$ and taking the Moore-Penrose pseudoinverse A^+ of A we define $\delta = A^+ e$ with $A\delta = e$ and $\|\delta\|_2 \leq \sigma_{\min}(A)^{-1}\|e\|_2$. Using this we can start to derive the dual bound by taking the difference of (4b) and (5b)

$$A^T(y - y^*) + (z - z^*) = Q(x - x^*) + g. \tag{8}$$

Multiplying from the left with $Q^{-1}(A^T(y - y^*) + (z - z^*))$ transposed gives

$$\begin{aligned}\|A^T(y - y^*) + (z - z^*)\|^2_{Q^{-1}} &= (A^T(y - y^*) + (z - z^*))^T((x - x^*) + Q^{-1}g)\\ &= (A^T(y - y^*) + (z - z^*))^T Q^{-1}g + (y - y^*)^T \underbrace{A(x - x^*)}_{A\delta} + (z - z^*)^T(x - x^*)\\ &= (A^T(y - y^*) + (z - z^*))^T(Q^{-1}g + \delta) + (z - z^*)^T(x - x^* - \delta).\end{aligned} \tag{9}$$

The second term of (9) can be expressed as

$$\begin{aligned}(z - z^*)^T(x - l - (x^* - l) - \delta) &= \underbrace{z^T(x - l)}_{i} + \underbrace{z^{*T}(x^* - l)}_{0} - \underbrace{z^T(x^* - l)}_{\geq h^T(x^* - l)} - \underbrace{z^{*T}(x - l)}_{\geq z^{*T} f}\\ (z - z^*)^T(x - l - (x^* - l) - \delta) &\leq i - h^T(x^* - l) - z^{*T} f.\end{aligned}$$

With this and (9) we bound from above the term $\|A^T(y - y^*) + (z - z^*)\|^2_{Q^{-1}} = *$ giving the inequality

$$* \leq (A^T(y - y^*) + (z - z^*))^T(Q^{-1}g + \delta) + i - h^T(x^* - l) - z^{*T} f.$$

Taking the norm on the right and reordering terms gives

$$\begin{aligned}\|Q\|_2^{-1}\|A^T(y - y^*) + (z - z^*)\|_2^2 &\leq \|A^T(y - y^*) + (z - z^*)\|_2\|Q^{-1}g + \delta\|_2\\ &+ \|i - h^T(x^* - l) - z^{*T} f\|_2.\end{aligned}$$

This is a quadratic expression in $\|A^T(y - y^*) + (z - z^*)\|_2 = m$

$$m^2 - m\|Q^{-1}g + \delta\|_2\|Q\|_2 - \|i - h^T(x^* - l) - z^{*T} f\|_2\|Q\|_2 \leq 0.$$

It has two roots, but only one is greater than zero and bounds $\|A^T(y - y^*) + (z - z^*)\|_2 (= m)$ from above

$$\begin{aligned}m \leq\ & \|Q^{-1}g + \delta\|_2\|Q\|_2/2\\ &+ \sqrt{(\|Q^{-1}g + \delta\|_2\|Q\|_2)^2/4 + \|i - h^T(x^* - l) - z^{*T} f\|_2\|Q\|_2}.\end{aligned} \tag{10}$$

This can be expressed as

$$\|A^T(y - y^*) + (z - z^*)\|_2 \leq a + \sqrt{a^2 + d} \tag{11}$$

where a and d are defined as above. This proves (6). To prove the primal bound we multiply equation (8) from the left with Q^{-1}

$$(x - x^*) = Q^{-1}(A^T(y - y^*) + (z - z^*) - g).$$

Taking norms gives the inequality

$$\|x - x^*\|_2 \leq \|Q^{-1}\|_2 \|A^T(y - y^*) + (z - z^*)\|_2 + \|Q^{-1}g\|_2. \tag{12}$$

Combining the dual bound (11) and (12) we get the final primal bound

$$\|x - x^*\|_2 \leq \lambda_{\min}(Q)(a + \sqrt{a^2 + d}) + \lambda_{\min}(Q)\|g\|_2$$

which proves (7). □

Note that $\lambda_{\max}(Q)\lambda_{\min}(Q)$ is the condition number of Q. The above assumption and lemmas can be summarized to a statement about the convergence of the algorithm for a strictly convex QP.

Theorem 9 (Rate of Convergence) *Algorithm 1 with corresponding input and using a QP solver satisfying Assumption 5 solving the QP (P) that is also strictly convex has a linear rate of convergence with a factor of $\tilde{\epsilon}^{1/2}$ for the primal iterates, i.e.*

$$\|x_k - x^*\| \leq \tilde{\epsilon}^{1/2}\|x_{k-1} - x^*\|,$$

with x^* being the unique solution of (P).

Proof By Assumption 5 and Lemma 6 we know that the right-hand side errors of the KKT conditions are bounded by

$$\|e\| \leq \tilde{\epsilon}^k,\ \|g\| \leq \tilde{\epsilon}^k,\ \|f\| \leq \tilde{\epsilon}^k,\ \|i\| \leq \sigma\tilde{\epsilon}^{2(k-1)},\ \|h\| = 0.$$

Here we set the violations h of the inequality KKT multipliers z to zero and count them as additional dual violations g for simplicity. Also note that in Lemma 8 the bound is just depending on the norm of the right-hand side violation vectors, two different violation vectors with the same norm give the same bound. Therefore we just consider the norms. Combining the above with Lemma 8 we get

$$\|x_k - x^*\| \leq c_1\tilde{\epsilon}^k + \sqrt{c_2\tilde{\epsilon}^k + c_3\tilde{\epsilon}^{2k}}$$

for the primal iterate in iteration k with constants

$$c_1 = \lambda_{\min}(Q)\lambda_{\max}(Q)\left(\frac{1}{\lambda_{\max}(Q)} + \frac{\lambda_{\min}(Q)}{2} + \frac{1}{2\sigma_{\min}(A)}\right)$$

$$c_2 = \lambda_{\max}(Q)\|z^*\|$$
$$c_3 = (c_1 - \lambda_{min}(Q))^2 + \lambda_{\max}(Q)\sigma/\tilde{\epsilon}^2.$$

Looking at the quotient

$$\frac{\|x_k - x^*\|}{\|x_{k-1} - x^*\|} \leq \frac{c_1\tilde{\epsilon}^k + \sqrt{c_2\tilde{\epsilon}^k + c_3\tilde{\epsilon}^{2k}}}{c_1\tilde{\epsilon}^{k-1} + \sqrt{c_2\tilde{\epsilon}^{k-1} + c_3\tilde{\epsilon}^{2(k-1)}}}$$

and seeing that

$$\frac{\|x_k - x^*\|}{\|x_{k-1} - x^*\|} \leq \frac{\tilde{\epsilon}^{k/2}(c_1\tilde{\epsilon}^{k/2} + \sqrt{c_2 + c_3\tilde{\epsilon}^k})}{\tilde{\epsilon}^{(k-1)/2}(c_1\tilde{\epsilon}^{(k-1)/2} + \sqrt{c_2 + c_3\tilde{\epsilon}^{k-1}})} = \tilde{\epsilon}^{1/2}\gamma_k$$

with $\gamma_k \leq 1$ proves the result. □

This theoretical investigation shows us two things. First, we have linear residual convergence with a rate of $\tilde{\epsilon}$. In contrast to usual convergence results our algorithm achieves this rate in practice by the use of rational computations if the floating-point solver delivers solutions of the quality specified in Assumption 5. This is also checked by the rational residual calculation in our algorithm in every iteration. Second, this residual convergence implies primal iterate convergence with a linear rate of $\tilde{\epsilon}^{1/2}$ for strictly convex QPs.

4 Implementation

Following previous work [10] on the LP case we implemented Algorithm 1 in the same framework within the `SoPlex` solver [24], version 2.2.1.2, using the GNU multiple precision library (GMP) [14] for rational computations, version 6.1.0. Note that `SoPlex` is not used to solve LPs but provides support functionalities. These are functionalities to read and write mps files (extended to qps files) and to save the corresponding QP problems in rational and floating-point precision. Additionally `SoPlex` provides rational and floating-point calculations with operator overloading reducing implementation complexity (based on GMP). As underlying QP solver we use the active-set solver `qpOASES` [4] version 3.2. This version of `qpOASES` was originally designed for small to medium QPs (up to 1000 variables and constraints). Furthermore, we implemented an interface to a pre-release version of `qpOASES` 4.0, which can handle larger, sparse QPs of a size up to 40,000 variables and constraints. Compared to the matured `qpOASES` 3.2, this version is not yet capable of hotstarts and in some cases is less robust. Nevertheless, it allows us to study the viability of iterative refinement on larger QPs. The source code of our implementation is available for download in a public repository [21].

In order to treat general QPs with inequalities, our implementation recovers the form (P) by adding one slack variable per inequality constraint. Note that not only

Table 1 IQPR parameters

Parameter set	s1	s2	s3	s4	s5
qpOASES version	3.2	3.2	4.0	4.0	3.2
Primal tolerance (ϵ_P)	1e−100	1e−100	1e−100	1e−100	1e−10
Dual tolerance (ϵ_D)	1e−100	1e−100	1e−100	1e−100	1e−10
Maxscaleincrement (α)	1e12	1e12	1e12	1e12	1e12
Sparse	No	No	Yes	Yes	No
Max num backstepping (l_{max})	10	10	10	10	1
Refinement limit (k_{max})	300	50	50	50	10
Ratfac minstalls	2	0	0	51	30

lower, but also upper bounds on the variables need to be considered. However, this is a straightforward modification to our algorithm and realized in the implementation.

One advantage of using the active-set QP solver qpOASES is the returned basis information. We use the basis in three aspects: first, to calculate dual and complementary slack violations; second, to explicitly set nonbasic variables to their lower bounds after the refinement step in *Line 16* of Algorithm 1; and third, to compute a rational solution defined by the corresponding system of linear equations. This is solved by a standard LU factorization in rational arithmetic. If the resulting primal-dual solution is verified to be primal and dual feasible, the algorithm can terminate early with an exact optimal basic solution.

Since the LU factorization can be computationally expensive, we only perform this step if we believe the basis to be optimal. When the QP solver returns the same basis as "optimal" for several iterations this can be used as a heuristic indicator that the basis might be truly optimal, even if the iteratively corrected numerical solution is not yet exact. Hence, the number of consecutive iterations with the same basis is used to trigger a rational basis system solve. This can be controlled by a threshold parameter called "ratfac minstalls", see Table 1.

If the floating-point solver fails to compute an approximately optimal solution, we decrease the scaling factor by two orders of magnitude and try to solve the resulting QP again. The scaling factor is reduced either until the maximum number of backstepping rounds is reached or until the next backstepping round would result in a scaling factor lower than in the last refinement iteration ($k - 1$).

The default parameter set (s1) of our implementation is given in Table 1. The other four parameter sets (s2–s5) are used for our numerical experiments to derive either exact or inexact solutions.

We exploit the different features of the two qpOASES versions. Version 3.2 has hotstart capabilities that allow reusing the internal basis system factorization of the preceding optimal basis. Therefore we start in the old optimal basis and build on the progress made in the previous iterations instead of solving the QP from scratch at every iteration. Additionally we increase the termination tolerance and relax other parameters that ensure a reliable solve. This speeds up the solving process and is possible because the inaccuracies, introduced by the floating-point solution, are detected anyway and handed back to the QP solver in the next iteration for correction. If the QP

Table 2 qpOASES options in Version 3.2

Option	Fast	Reliable
Standard settings set	MPC	Reliable
NZCTests	Enabled	Enabled (default)
DriftCorrection	Enabled	Enabled (default)
Ramping	Enabled	Enabled (default)
terminationTolerance	1e−3	1.1105e−9 (default)
numRefinementSteps	0 (default)	10
enableFullLITests	0	0

solver fails we simply change to reliable settings and resolve the same QP from the same starting basis before downscaling. Hence, in Algorithm 1 each 'solve' statement means: try fast settings first and if this fails switch to slow and reliable settings of qpOASES 3.2. These two sets of options are given in Table 2. In this Table we only state the options chosen differently from the standard qpOASES settings sets (MPC, Reliable) which are given in the "Appendix" in Table 5.

For the pre-release version 4.0 we use default settings and no resolves. We either factorize after each iteration or not at all (see Table 1).

5 Numerical results

For the numerical experiments the standard testset of Maros and Mészáros [19] was used. It contains 138 convex QPs that feature between two and about 90,000 variables. The number of constraints varies from one to about 180,000 and the number of nonzeros ranges between two and about 550,000. The computations were performed on a cluster of 64-bit Intel XeonE5-2660 (v3) CPUs at 2.6 GHz with 25 MB L3 cache and 125 GB main memory.

We conduct two different experiments. The goal of the first experiment is to solve as many QPs from the testset as precisely as possible in order to analyze the iterative refinement procedure computationally and to provide exact reference solutions for future research on QP solvers. In the second experiment we want to compare qpOASES (version 3.2, no QP refinement, one solve, default settings) to low accuracy refinement (low tolerance of 1e−10 in Algorithm 1, using also qpOASES 3.2). This allows us to investigate whether refinement could also be beneficial in cases that do not require extremely high accuracy, but a strictly guaranteed solution tolerance in shortest possible runtime.

Experiment 1 We use the three different parameter sets (s2–s4) given in Table 1 to calculate exact solutions. The first set (s2) contains a primal and dual termination tolerance of 1e−100, enables rational factorization in every iteration, and allows for 50 refinements and 10 backsteppings using a dense QP formulation with qpOASES version 3.2. In contrast the other two sets (s3, s4) with qpOASES version 4.0 use a sparse QP formulation, either with factorization in every iteration or without factorization. For this experiment, a time limit of three hours is imposed per instance and solver.

Table 3 Results for the three exact parameter sets (s2–s4) over all 138 QPs in the testset: number of instances according to terminal solution accuracy for each setting, for the virtual best setting, and the average number of nonzeros over the instances in the "best" categories

Accuracy reached	s2	s3	s4	best	avg. nnzs
Exact (no viol.)	73	74	1	91	6.76e+03
High ($\leq$ 1e−100)	33	45	118	39	1.45e+04
Low ($>$ 1e−100)	11	18	18	8	9.34e+04
Fail (not returned)	21	1	1	0	

Table 3 states for each setting the number of instances which were solved exactly, for which tolerance 1e−100 was reached, for which only low tolerance solutions were produced, and the number of instances which did not return from the cluster due to memory limitations. In total these three strategies could solve 91 out of the 138 QPs in the testset *exactly* and 39 instances within tolerance 1e−100. For eight instances no high-precision solution was computed. These "virtual best" results stated in the fifth column consider for each QP the result of the individual parameter sets that resulted in the smallest violation. It should be emphasized that for each of the three parameter sets there exists at least one instance for which it produced the most accurate solution.

The last column reports the average number of nonzeros of the QPs in the three "virtual best" categories. This suggests that for problems with fewer nonzeros a higher accuracy was reached. In the third and fourth column one problem did not return (BOYD1 with over 90000 variables). For the parameter set s4 without rational factorization we see that one QP is solved exactly while for all others the algorithm terminates with violations greater zero.

In order to solve the 197 ($= 33 + 45 + 118$) QPs to high precision the algorithm needed on average 8.84 refinements. This confirms the linear convergence because we bounded the increase of the scaling factor in each iteration by α = 10e+12 and terminate after reaching a tolerance of 1e−100. If `qpOASES` would consistently return solutions with an accuracy of 1e−12 we would expect the algorithm to need 9 iterations ($100/12 \approx 8.33\ldots$ rounded up). We see that `qpOASES` usually delivers solutions of a tolerance below 1e−12.

Detailed results can be found in the "Appendix" in Tables 6, 7, and 8. The column "Status" reports "optimal" if a solution of tolerance below 1e−100 is computed. Otherwise, if the tolerance is not reached the status is declared "fail". If the objective value found differs from the value in literature [19] by more than 1e−7, then the status is reported as "inconsistent".[1] If we exceed the timelimit, then the status is "timeout". The status "error" is an internal algorithmic error, e.g., the QP solver fails to solve one of the QPs in the sequence and hence the algorithm stops. If the maximum number of algorithm iterations is reached (e.g., because the solutions calculated by the QP solver violate Assumption 5) the status is "abort" and results in "NaN" (not a number) in the other columns. The column "Iterations" counts all QP solver iterations (active set changes) summed over all algorithm iterations. The algorithm iterations are the

[1] For 5 QP problems our solution is more precise using parameter set 2, for parameter sets 3 and 4 this are 6 QP problems. For zero problems our solution is not precise with the second parameter set while the sets 3 and 4 produce 4 and 7 imprecise solutions.

number of refinements (plus one) given in the "Refinement" column. The QP solver iterations were only counted for the parameter set s2 and hence are zero for the other two settings.

If an exact solution is found qpOASES usually returns the optimal basis in the first three refinement iterations. The optimal basis was found in the first iteration (without refinement) for 55 instances when using parameter set s2, with set s3 and s4 this where 74 and one instances. Subsequently, the corresponding basis system is solved exactly by a rational LU factorization.[2] For six problems we found that the objective values given in [19] differ from our results by more than 1e−7: GOULDQP2, HS268, S268, HUESTIS, HUES-MOD, and LISWET8. This might be due to the use of a floating-point QP solver with termination tolerance about 1e−7 when originally computing the values reported. The precise objective value can be found in the online material associated with this paper.

Experiment 2 In the following the iterative refinement algorithm is set to a termination tolerance 10^{-10} and the rational factorization of the basis system is disabled. The refinement limit is set to 10 and the backstepping limit is set to one (parameter set s5). We compare this implementation to qpOASES 3.2 with the three predefined qpOASES settings (MPC, Default, Reliable) that include termination tolerances of 2.2210e−7, 1.1105e−9, and 1.1105e−9, respectively. For these fast solves we select only part of the testset, including the 73 problems that have no more than 1,000 variables and constraints. This corresponds to the problem size for which qpOASES 3.2 was originally designed. In order to allow for a meaningful comparison of runtimes, the evaluation only considers QPs which were solved by all three qpOASES 3.2 settings and by refinement to "optimality", where optimality was certified by qpOASES 3.2 (with its internal floating-point checks) or rational checks in our algorithm, respectively. For this experiment, a time limit of one hour is imposed per instance and solver.

An overview of the performance results is given in Table 4. We report runtime, QP solver iterations, and the final tolerance reached, each time as arithmetic and shifted geometric mean. To facilitate a more detailed analysis, we consider the series of subsets "$> t$" of instances, for which at least one algorithm took more than t seconds. Equivalently, we exclude the QPs for which all settings took at most 0.01 s, 0.1 s, 1 s, and 10 s. Defining the exclusion by all instead of one method only avoids a biased definition of these sets of increasing difficulty.

The results show that in no case is the mean runtime of the refinement algorithm larger than the runtime of qpOASES with reliable setting. At the same time, the accuracy reached is always significantly higher. Compared to qpOASES Default, which results in an even lower level of precision, refinement is faster in arithmetic and slightly slower in shifted geometric mean. The QP solver iterations of the refinement are comparable to the MPC setting. When looking at the different subsets we see that for QPs with larger runtime the refinement approach performs relatively better (smaller runtime, iterations and lower tolerance) than the three qpOASES 3.2 standard settings. The refinement guarantees the tolerance of 1e−10 if it does not fail. To achieve this

[2] This is not the case for the parameter set s4 where we disabled this option. Nevertheless an exact primal and dual solution was found by the QP solver for one instance (HS21).

Table 4 Performance comparison for inexact solves (runtimes are in seconds)

Measure	Subset	IQPR s5		qpOASES with standard settings		
				MPC	Default	Reliable
Time: arith. mean (% rat. time)	All	2.54	(0.16)	1.03	2.77	19.58
	> 0.01	3.25	(0.16)	1.32	3.55	25.08
	> 0.1	4.02	(0.12)	1.64	4.40	31.07
	> 1	5.66	(0.12)	2.31	6.27	44.60
	> 10	7.19	(0.11)	2.77	9.06	69.39
Time: shifted geo. mean, shift = 0.01 (% rat. time)	All	0.16	(1.68)	0.08	0.10	0.16
	> 0.01	0.36	(0.98)	0.15	0.20	0.36
	> 0.1	0.60	(0.54)	0.24	0.32	0.64
	> 1	0.94	(0.49)	0.43	0.67	1.66
	> 10	0.54	(1.00)	0.26	0.51	1.51
QP solver iterations: arith. mean	All	283.75		260.53	389.92	386.96
	> 0.01	362.16		332.44	496.91	493.09
	> 0.1	436.67		400.96	591.35	586.96
	> 1	520.91		479.38	765.59	761.56
	> 10	353.00		348.25	837.15	832.75
QP solver iterations: shifted geo. mean, shift = 1	All	38.43		36.86	62.08	61.68
	> 0.01	85.18		80.86	113.98	112.72
	> 0.1	108.12		101.95	124.46	123.05
	> 1	105.41		100.22	144.20	142.88
	> 10	36.21		35.23	69.62	69.39
Tolerance: arith. mean	All	1.49e−12		1.29e−08	1.10e−08	2.28e−09
	> 0.01	1.91e−12		1.65e−08	1.40e−08	2.92e−09
	> 0.1	2.10e−12		2.03e−08	1.74e−08	3.62e−09
	> 1	8.89e−13		2.21e−08	2.48e−08	4.98e−09
	> 10	5.29e−14		1.42e−08	3.92e−08	7.32e−09
Tolcerance: shifted geo. mean, shift = 1e−20	All	1.29e−16		2.14e−12	1.62e−15	4.34e−15
	> 0.01	8.71e−17		1.00e−11	4.91e−15	1.13e−14
	> 0.1	3.57e−17		8.45e−12	6.92e−15	2.10e−14
	> 1	1.94e−17		1.08e−12	2.44e−15	6.02e−15
	> 10	9.36e−19		6.86e−15	3.21e−16	2.45e−16

tolerance, for 9 QPs two refinements were necessary, for 21 QPs only one refinement was necessary, and for 35 instances no refinement was necessary at all. The rational computation overhead stated in brackets after the runtime and is well below 2%. The details are shown in Table 9 in the "Appendix". Also note that due to exclusion of fails

(which mainly occur with the qpOASES MPC settings) the summarized results have a slight bias towards qpOASES.

6 Conclusion

We presented a novel refinement algorithm and proved linear convergence of residuals and errors. Notably, this theoretical convergence result also carries over to our implementation due to the use of exact rational calculations. We provided high-precision solutions for most of the QPs in the Maros and Mészáros testset, correcting inaccuracies in optimal solution values reported in the literature. This is beneficial for future research on QP solvers that are evaluated on this testset.

In a second experiment we saw that iterative refinement provides proven tolerance solutions with smaller or equal computation times compared to qpOASES with "Reliable" solver settings. It can therefore be used as a tool to increase the reliability and speed of standard floating-point QP solvers. The related approach in [18] is designed to speed up QP solutions without extended-precision or rational arithmetic. One could think of combining the two approaches, only using extended precision when necessary, e.g., when convergence stalls.

If optimal solutions are needed for rigorous reasoning or to make decisions in the real world the algorithm presented is useful because it is able to fully ensure a specified tolerance. This tolerance then can be adapted to the necessity of the application at hand. At the same time this comes with little overhead in rational computation time, which is important for practical applications.

Regarding algorithmic research and solver development, our framework also provides the possibility to compare different floating-point QP solvers by looking at the number of refinements needed with each solver to detect optimal bases or solutions of a specified tolerance as a measure for solver accuracy. Solver robustness can be checked precisely because violations are computed in rational precision. In the future, it would be valuable to extend the implementation to handle cases of unbounded or infeasible QPs and to experiment with more general variable transformations that apply, e.g., a different scaling factor for each variable. As a final remark, we hope that the idea of checking numerical results of floating-point algorithms in exact or safe arithmetic will become a future trend when applying or analyzing numerical algorithms.

7 Appendix

The following tables contain a list of detailed parameter settings for the QP solver qpOASES and instancewise results for the different experiments described in Sect. 5.

7.1 Detailed QP solver options

See Table 5.

Table 5 Standard parameter values for different qpOASES 3.2 settings (EPS = 2.221e−16)

Option	Default	Reliable	MPC
enableRamping	Enabled	Enabled	Disabled
enableFarBounds	Enabled	Enabled	Enabled
enableFlippingBounds	Enabled	Enabled	Disabled
enableRegularisation	Disabled	Disabled	Enabled
enableFullLITests	Disabled	Enabled	Disabled
enableNZCTests	Enabled	Enabled	Disabled
enableDriftCorrection	1	1	0
enableCholeskyRefactorisation	0	1	0
enableEqualities	Disabled	Disabled	Enabeld
terminationTolerance	5.0e6 * EPS	5.0e6 * EPS	1.0e9 * EPS
boundTolerance	1.0e6 * EPS	1.0e6 * EPS	1.0e6 * EPS
boundRelaxation	1.0e4	1.0e4	1.0e4
epsNum	− 1.0e3 * EPS	− 1.0e3 * EPS	− 1.0e3 * EPS
epsDen	1.0e3 * EPS	1.0e3 * EPS	1.0e3 * EPS
maxPrimalJump	1.0e8	1.0e8	1.0e8
maxDualJump	1.0e8	1.0e8	1.0e8
initialRamping	0.5	0.5	0.5
finalRamping	1.0	1.0	1.0
initialFarBounds	1.0e6	1.0e6	1.0e6
growFarBounds	1.0e3	1.0e3	1.0e3
initialStatusBounds	Lower	Lower	Inactive
epsFlipping	1.0e3 * EPS	1.0e3 * EPS	1.0e3 * EPS
numRegularisationSteps	0	0	1
epsRegularisation	1.0e3 * EPS	1.0e3 * EPS	1.0e3 * EPS
numRefinementSteps	1	2	0
epsIterRef	1.0e2 * EPS	1.0e2 * EPS	1.0e2 * EPS
epsLITests	1.0e5 * EPS	1.0e5 * EPS	1.0e5 * EPS
epsNZCTests	3.0e3 * EPS	3.0e3 * EPS	3.0e3 * EPS
enableDropInfeasibles	Disabled	Disabled	Disabled
dropBoundPriority	1	1	1
dropEqConPriority	1	1	1
dropIneqConPriority	1	1	1
enableInertiaCorrection	Enabled	Enabled	Enabled
rcondSMin	1.0e−14	1.0e−14	1.0e−14

7.2 Detailed results

See Tables 6, 7, 8 and 9.

Table 6 Detailed results for exact solve of large QP set with parameter set s2

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
AUG3D	Optimal	3492.32	0	2.56e−113	8	0	0
AUG3DC	Optimal	219.36	0	00	0	0	0
AUG3DCQP	Optimal	218.64	540	00	1	0	0
AUG3DQP	Optimal	3092.52	324	2.15e−110	8	0	0
BOYD1	Abort	NaN	NaN	NaN	NaN	NaN	NaN
BOYD2	Abort	NaN	NaN	NaN	NaN	NaN	NaN
CONT-050	Optimal	4306.67	1	00	0	0	0
CONT-100	Timeout	10814.69	3	NaN	7	6	0
CONT-101	Timeout	10830.68	2	NaN	7	6	0
CONT-300	Abort	NaN	NaN	NaN	NaN	NaN	NaN
CVXQP1_L	Fail	15786.74	4010	55500	0	0	0
CVXQP1_M	Optimal	16.78	412	00	0	0	0
CVXQP1_S	Optimal	0.04	36	00	0	0	0
CVXQP2_M	Optimal	25.97	651	00	0	0	0
CVXQP2_S	Optimal	0.01	62	00	0	0	0
CVXQP3_L	Timeout	10803.72	2990	NaN	1	0	0
CVXQP3_M	Optimal	5.95	231	00	1	0	0
CVXQP3_S	Optimal	0.00	22	00	0	0	0
DPKLO1	Optimal	2.23	0	00	0	0	0
DTOC3	Fail	19121.42	0	4.40e−15	0	0	0
DUAL1	Optimal	0.26	26	00	1	0	0
DUAL2	Optimal	1.05	4	00	1	0	0
DUAL3	Optimal	1.90	14	00	1	0	0
DUAL4	Optimal	0.21	13	00	0	0	0
DUALC1	Optimal	0.11	31	00	0	0	0
DUALC2	Optimal	0.07	28	00	0	0	0
DUALC5	Optimal	0.07	3	00	0	0	0
DUALC8	Optimal	0.39	13	00	0	0	0
EXDATA	Optimal	8964.52	6738	00	1	0	0
GENHS28	Optimal	0.00	0	00	0	0	0
GOULDQP2	Inconsistent	6.70	585	00	2	0	0
GOULDQP3	Optimal	1.84	176	00	0	0	0
HS118	Optimal	0.00	24	00	0	0	0
HS21	Optimal	0.00	3	00	0	0	0
HS268	Inconsistent	0.00	0	00	0	0	0

Table 6 continued

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
HS35	Optimal	0.00	1	00	0	0	0
HS35MOD	Optimal	0.00	0	00	0	0	0
HS51	Optimal	0.00	0	00	0	0	0
HS52	Optimal	0.00	0	00	0	0	0
HS53	Optimal	0.00	0	00	0	0	0
HS76	Optimal	0.00	4	00	0	0	0
HUESTIS	Inconsistent	9782.42	554	00	0	0	0
HUES-MOD	Inconsistent	9408.64	554	00	0	0	0
KSIP	Optimal	31.56	1080	00	1	0	0
LASER	Optimal	593.20	2838	00	0	0	0
LOTSCHD	Optimal	0.00	5	00	0	0	0
MOSARQP1	Optimal	445.13	1528	00	0	0	0
MOSARQP2	Optimal	21.45	332	00	0	0	0
PRIMAL1	Optimal	0.51	77	00	0	0	0
PRIMAL2	Optimal	2.66	94	00	0	0	0
PRIMAL3	Optimal	4.37	117	00	0	0	0
PRIMAL4	Optimal	21.04	88	00	0	0	0
PRIMALC1	Optimal	0.11	234	00	0	0	0
PRIMALC2	Optimal	0.17	237	00	0	0	0
PRIMALC5	Optimal	0.23	288	00	0	0	0
PRIMALC8	Optimal	1.86	515	00	0	0	0
Q25FV47	Optimal	372.89	7362	00	0	0	0
QADLITTL	Optimal	0.09	232	00	0	0	0
QAFIRO	Optimal	0.00	30	6.15e−107	7	0	0
QBANDM	Optimal	2.93	1164	00	0	0	0
QBEACONF	Optimal	0.91	305	8.07e−110	9	0	0
QBORE3D	Optimal	0.87	536	1.97e−111	9	0	0
QBRANDY	Optimal	1.07	450	4.86e−108	8	0	0
QCAPRI	Optimal	2.95	1051	1.76e−106	8	0	1
QE226	Optimal	4.66	1396	1.71e−101	8	0	0
QETAMACR	Optimal	4.75	559	5.17e−102	9	0	0
QFFFFF80	Error	44.78	1079	NaN	9	3	5
QFORPLAN	Optimal	2.20	1174	00	0	0	0
QGFRDXPN	Optimal	29.31	1423	00	0	0	0
QGROW15	Optimal	8.11	632	2.54e−105	8	0	0
QGROW22	Optimal	32.92	944	2.86e−109	10	0	0
QGROW7	Optimal	0.83	298	7.99e−112	8	0	0
QISRAEL	Optimal	0.38	290	00	0	0	0
QPCBLEND	Optimal	0.06	66	00	1	0	0

Table 6 continued

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
QPCBOEI1	Optimal	3.64	435	00	1	0	0
QPCBOEI2	Optimal	0.23	175	00	0	0	0
QPCSTAIR	Optimal	2.76	257	00	0	0	0
QPILOTNO	Optimal	573.39	7442	1.99e−102	13	0	0
QPTEST	Optimal	0.00	1	00	0	0	0
QRECIPE	Optimal	0.07	48	9.74e−110	7	0	0
QSC205	Optimal	0.78	106	1.50e−107	8	0	0
QSCAGR25	Optimal	6.17	1076	00	0	0	0
QSCAGR7	Optimal	0.14	329	00	0	0	0
QSCFXM1	Optimal	8.05	801	1.13e−112	11	0	1
QSCFXM2	Optimal	81.35	1891	1.25e−108	10	0	0
QSCFXM3	Optimal	319.56	2510	1.77e−107	10	0	0
QSCORPIO	Optimal	2.43	233	1.08e−101	7	0	0
QSCRS8	Optimal	49.19	2016	00	1	0	0
QSCSD1	Optimal	8.78	2054	00	1	0	0
QSCSD6	Optimal	106.56	6015	4.11e−101	9	0	0
QSCSD8	Optimal	1560.58	19564	2.22e−104	8	0	0
QSCTAP1	Optimal	11.22	1474	2.63e−103	8	0	0
QSCTAP2	Optimal	860.50	3862	6.58e−113	10	0	0
QSCTAP3	Optimal	2156.04	6964	2.49e−101	8	0	0
QSEBA	Optimal	26.60	1677	00	0	0	0
QSHARE1B	Optimal	0.51	782	00	1	0	0
QSHARE2B	Optimal	0.19	359	00	1	0	0
QSHELL	Optimal	119.14	3306	7.69e−109	9	0	1
QSHIP04L	Optimal	398.29	4847	1.67e−106	8	0	0
QSHIP04S	Optimal	130.24	2908	2.11e−107	8	0	1
QSHIP08L	Optimal	3711.96	7911	3.68e−102	8	0	0
QSHIP08S	Optimal	738.69	3997	1.86e−110	9	0	0
QSHIP12L	Optimal	8386.98	9765	4.22e−109	8	0	1
QSHIP12S	Optimal	1176.10	3821	2.30e−109	8	0	0
QSIERRA	Error	996.02	6421	NaN	0	0	1
QSTAIR	Optimal	13.43	1136	7.18e−111	9	1	2
QSTANDAT	Optimal	26.45	1222	00	3	0	0
S268	Inconsistent	0.00	0	00	0	0	0
STADAT1	Optimal	8013.74	9995	00	0	0	0
STADAT2	Optimal	6437.65	6527	00	1	0	0
STADAT3	Timeout	18337.54	2144	NaN	0	0	1
STCQP1	Optimal	604.13	353	1.05e−103	7	0	0
STCQP2	Optimal	419.60	105	00	0	0	0

Table 6 continued

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
TAME	Optimal	0.00	0	00	0	0	0
VALUES	Optimal	0.10	143	00	2	0	1
YAO	Optimal	659.58	2001	00	1	0	0
ZECEVIC2	Optimal	0.00	3	00	0	0	0

Iter. iterations, *Tol.* tolerance, *Ref.* refinements, *Back.* backstepping, *Res.* resolves

Table 7 Detailed results for exact solve of large QP set with parameter set s3

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
AUG2D	Timeout	10865.56	0	NaN	1	0	0
AUG2DC	Timeout	10836.45	0	NaN	6	5	0
AUG2DCQP	Optimal	155.71	0	1.47e−101	8	0	0
AUG2DQP	Timeout	10801.84	0	NaN	6	5	0
AUG3D	Error	7.60	0	NaN	1	0	0
AUG3DC	Optimal	112.66	0	00	0	0	0
AUG3DCQP	Optimal	19.50	0	00	0	0	0
AUG3DQP	Optimal	7.62	0	00	0	0	0
BOYD2	Abort	NaN	NaN	NaN	NaN	NaN	NaN
CONT-050	Optimal	4224.16	0	00	0	0	0
CONT-100	Timeout	10806.77	0	NaN	7	6	0
CONT-101	Timeout	10809.28	0	NaN	7	6	0
CONT-200	Timeout	10802.04	0	NaN	7	6	0
CONT-201	Timeout	10801.56	0	NaN	7	6	0
CONT-300	Abort	NaN	NaN	NaN	NaN	NaN	NaN
CVXQP1_L	Optimal	6641.26	0	1.90e−101	9	0	0
CVXQP1_M	Optimal	8.02	0	00	0	0	0
CVXQP1_S	Optimal	0.02	0	00	0	0	0
CVXQP2_L	Timeout	11041.07	0	NaN	5	4	0
CVXQP2_M	Optimal	19.13	0	00	0	0	0
CVXQP2_S	Optimal	0.03	0	2.25e−110	7	0	0
CVXQP3_L	Error	7587.33	0	NaN	6	1	0
CVXQP3_M	Optimal	2.33	0	00	0	0	0
CVXQP3_S	Reached	0.48	0	NaN	50	0	0
DPKLO1	Optimal	2.26	0	00	0	0	0
DTOC3	Optimal	2611.08	0	00	0	0	0
DUAL1	Optimal	0.15	0	00	0	0	0

Table 7 continued

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
DUAL2	Optimal	0.53	0	00	0	0	0
DUAL3	Optimal	0.78	0	00	0	0	0
DUAL4	Optimal	0.20	0	00	0	0	0
DUALC1	Optimal	0.07	0	00	0	0	0
DUALC2	Optimal	0.05	0	00	0	0	0
DUALC5	Optimal	0.08	0	00	0	0	0
DUALC8	Optimal	0.19	0	00	0	0	0
EXDATA	Optimal	1407.82	0	00	0	0	0
GENHS28	Optimal	0.00	0	00	0	0	0
GOULDQP2	Inconsistent	0.24	0	1.50e−110	7	0	0
GOULDQP3	Optimal	0.11	0	00	0	0	0
HS118	Optimal	0.00	0	00	0	0	0
HS21	Optimal	0.00	0	00	0	0	0
HS268	Inconsistent	0.00	0	00	0	0	0
HS35	Optimal	0.00	0	00	0	0	0
HS35MOD	Optimal	0.00	0	00	0	0	0
HS51	Optimal	0.00	0	3.14e−108	6	0	0
HS52	Optimal	0.00	0	00	0	0	0
HS53	Optimal	0.00	0	00	0	0	0
HS76	Optimal	0.00	0	00	0	0	0
HUESTIS	Inconsistent	40.17	0	00	0	0	0
HUES-MOD	Inconsistent	21.27	0	00	0	0	0
KSIP	Optimal	2.38	0	00	0	0	0
LASER	Optimal	41.47	0	00	0	0	0
LISWET1	Optimal	57.85	0	00	0	0	0
LISWET10	Optimal	645.20	0	1.60e−105	15	0	0
LISWET11	Optimal	54.26	0	00	0	0	0
LISWET12	Optimal	63.22	0	00	0	0	0
LISWET2	Optimal	704.57	0	4.17e−107	17	0	0
LISWET3	Optimal	306.45	0	1.20e−101	14	0	0
LISWET4	Optimal	317.64	0	2.60e−107	15	0	0
LISWET5	Optimal	324.42	0	4.27e−106	15	0	0
LISWET6	Optimal	325.27	0	3.68e−102	14	0	0
LISWET7	Optimal	54.87	0	00	0	0	0
LISWET8	Inconsistent	56.98	0	00	0	0	0
LISWET9	Optimal	64.09	0	00	0	0	0

Table 7 continued

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
LOTSCHD	Optimal	0.00	0	00	0	0	0
MOSARQP1	Optimal	0.96	0	00	0	0	0
MOSARQP2	Optimal	0.52	0	00	0	0	0
POWELL20	Optimal	44.65	0	00	0	0	0
PRIMAL1	Optimal	0.20	0	00	0	0	0
PRIMAL2	Optimal	0.71	0	00	0	0	0
PRIMAL3	Optimal	1.45	0	00	0	0	0
PRIMAL4	Optimal	0.76	0	00	0	0	0
PRIMALC1	Optimal	0.01	0	00	0	0	0
PRIMALC2	Optimal	0.01	0	00	0	0	0
PRIMALC5	Optimal	0.01	0	00	0	0	0
PRIMALC8	Optimal	0.02	0	00	0	0	0
Q25FV47	Optimal	9.14	0	3.75e−114	9	0	0
QADLITTL	Optimal	0.01	0	00	0	0	0
QAFIRO	Optimal	0.00	0	00	0	0	0
QBANDM	Optimal	0.20	0	1.17e−103	6	0	0
QBEACONF	Inconsistent	0.16	0	NaN	50	0	0
QBORE3D	Inconsistent	0.18	0	NaN	50	0	0
QBRANDY	Optimal	0.18	0	4.04e−106	9	0	0
QCAPRI	Optimal	0.11	0	00	0	0	0
QE226	Optimal	0.19	0	4.36e−107	8	0	0
QETAMACR	Optimal	1.24	0	3.85e−106	10	0	0
QFFFFF80	Optimal	0.99	0	3.05e−114	9	0	0
QFORPLAN	Optimal	0.23	0	1.88e−113	8	0	0
QGFRDXPN	Optimal	0.45	0	3.34e−104	9	0	0
QGROW15	Optimal	0.37	0	00	0	0	0
QGROW22	Optimal	0.90	0	00	0	0	0
QGROW7	Optimal	0.22	0	00	0	0	0
QISRAEL	Optimal	0.11	0	00	0	0	0
QPCBLEND	Optimal	0.03	0	00	0	0	0
QPCBOEI1	Optimal	0.58	0	5.70e−114	10	0	0
QPCBOEI2	Optimal	0.10	0	00	0	0	0
QPCSTAIR	Optimal	0.96	0	1.04e−109	11	0	0
QPILOTNO	Inconsistent	40.49	0	NaN	50	0	0
QPTEST	Optimal	0.00	0	00	0	0	0
QRECIPE	Inconsistent	0.04	0	NaN	50	0	0
QSC205	Optimal	0.30	0	1.27e−107	11	0	0
QSCAGR25	Optimal	0.32	0	1.06e−104	7	0	0

Table 7 continued

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
QSCAGR7	Optimal	0.02	0	3.71e−107	8	0	0
QSCFXM1	Optimal	0.20	0	1.79e−112	7	0	0
QSCFXM2	Optimal	0.73	0	5.01e−112	9	0	0
QSCFXM3	Optimal	1.46	0	3.59e−101	8	0	0
QSCORPIO	Optimal	0.16	0	2.31e−101	8	0	0
QSCRS8	Optimal	0.36	0	1.58e−108	13	0	0
QSCSD1	Optimal	0.12	0	7.72e−110	7	0	0
QSCSD6	Optimal	0.18	0	2.69e−111	7	0	0
QSCSD8	Optimal	1.16	0	00	0	0	0
QSCTAP1	Optimal	0.18	0	6.15e−106	7	0	0
QSCTAP2	Optimal	0.55	0	5.86e−111	6	0	0
QSCTAP3	Optimal	1.04	0	00	0	0	0
QSEBA	Optimal	0.24	0	1.15e−103	9	0	0
QSHARE1B	Optimal	0.07	0	00	0	0	0
QSHARE2B	Optimal	0.04	0	00	0	0	0
QSHELL	Optimal	0.82	0	3.16e−103	9	0	0
QSHIP04L	Optimal	0.60	0	7.13e−110	9	0	0
QSHIP04S	Optimal	0.43	0	5.88e−109	9	0	0
QSHIP08L	Optimal	2.14	0	2.31e−110	7	0	0
QSHIP08S	Optimal	1.21	0	2.31e−108	9	0	0
QSHIP12L	Optimal	2.78	0	3.71e−113	8	0	0
QSHIP12S	Optimal	3.00	0	2.82e−104	9	0	0
QSIERRA	Error	0.95	0	NaN	6	0	0
QSTAIR	Optimal	0.58	0	4.32e−101	8	0	0
QSTANDAT	Optimal	0.06	0	3.62e−107	7	0	0
S268	Inconsistent	0.00	0	00	0	0	0
STADAT1	Optimal	9.82	0	00	0	0	0
STADAT2	Optimal	9.89	0	00	0	0	0
STADAT3	Optimal	40.23	0	00	0	0	0
STCQP1	Optimal	47.09	0	1.91e−107	11	0	0
STCQP2	Optimal	25.77	0	5.65e−102	9	0	0
TAME	Optimal	0.00	0	00	0	0	0
UBH1	Optimal	136.35	0	00	0	0	0
VALUES	Optimal	0.03	0	00	0	0	0
YAO	Optimal	1.53	0	00	0	0	0
ZECEVIC2	Optimal	0.00	0	00	0	0	0

Iter. iterations, *Tol.* tolerance, *Ref.* refinements, *Back.* backstepping, *Res.* resolves

Table 8 Detailed results for exact solve of large QP set with parameter set s4

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
AUG2D	Error	55.14	0	NaN	1	0	0
AUG2DC	Optimal	56.77	0	4.76e−109	7	0	0
AUG2DCQP	Optimal	154.49	0	1.47e−101	8	0	0
AUG2DQP	Optimal	320.54	0	3.10e−107	7	0	0
AUG3D	Error	1.54	0	NaN	1	0	0
AUG3DC	Optimal	6.38	0	7.59e−108	7	0	0
AUG3DCQP	Optimal	4.73	0	1.52e−102	6	0	0
AUG3DQP	Optimal	3.31	0	8.00e−103	6	0	0
BOYD2	Abort	NaN	NaN	NaN	NaN	NaN	NaN
CONT-050	Optimal	53.31	0	2.25e−108	8	0	0
CONT-100	Optimal	1076.87	0	2.86e−102	7	0	0
CONT-101	Optimal	1029.04	0	5.89e−101	7	0	0
CONT-200	Timeout	13506.53	0	NaN	4	0	0
CONT-201	Timeout	11449.02	0	NaN	3	0	0
CONT-300	Abort	NaN	NaN	NaN	NaN	NaN	NaN
CVXQP1_L	Optimal	6853.92	0	1.90e−101	9	0	0
CVXQP1_M	Optimal	5.74	0	4.34e−107	9	0	0
CVXQP1_S	Optimal	0.05	0	8.55e−102	8	0	0
CVXQP2_L	Optimal	550.82	0	9.00e−109	9	0	0
CVXQP2_M	Optimal	0.98	0	1.12e−108	9	0	0
CVXQP2_S	Optimal	0.02	0	2.25e−110	7	0	0
CVXQP3_L	Error	7293.07	0	NaN	6	1	0
CVXQP3_M	Optimal	18.06	0	3.73e−111	9	0	0
CVXQP3_S	Reached	0.45	0	NaN	50	0	0
DPKLO1	Optimal	0.09	0	8.39e−108	9	0	0
DTOC3	Optimal	112.18	0	1.48e−114	7	0	0
DUAL1	Optimal	0.07	0	3.21e−103	10	0	0
DUAL2	Optimal	0.06	0	1.54e−101	6	0	0
DUAL3	Optimal	0.06	0	2.48e−103	6	0	0
DUAL4	Optimal	0.04	0	1.37e−104	6	0	0
DUALC1	Optimal	0.05	0	8.05e−106	6	0	0
DUALC2	Optimal	0.07	0	1.44e−106	6	0	0
DUALC5	Optimal	0.06	0	6.55e−107	6	0	0
DUALC8	Optimal	0.22	0	7.30e−105	6	0	0
EXDATA	Error	560.71	0	NaN	5	0	0
GENHS28	Optimal	0.00	0	1.25e−110	6	0	0
GOULDQP2	Inconsistent	0.22	0	1.50e−110	7	0	0
GOULDQP3	Optimal	0.36	0	4.48e−108	11	0	0

Table 8 continued

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
HS118	Inconsistent	0.02	0	NaN	50	0	0
HS21	Optimal	0.00	0	00	0	0	0
HS268	Inconsistent	0.00	0	6.35e−112	9	0	0
HS35	Optimal	0.00	0	9.00e−106	6	0	0
HS35MOD	Optimal	0.00	0	4.03e−108	6	0	0
HS51	Optimal	0.00	0	3.14e−108	6	0	0
HS52	Optimal	0.00	0	4.97e−106	6	0	0
HS53	Optimal	0.00	0	2.34e−108	6	0	0
HS76	Optimal	0.00	0	3.83e−108	6	0	0
HUESTIS	Inconsistent	37.99	0	6.05e−104	7	0	0
HUES-MOD	Inconsistent	18.39	0	4.21e−108	7	0	0
KSIP	Optimal	2.11	0	3.89e−105	6	0	0
LASER	Optimal	2.94	0	7.30e−105	12	0	0
LISWET1	Optimal	768.27	0	1.01e−107	15	0	0
LISWET10	Optimal	638.05	0	1.60e−105	15	0	0
LISWET11	Optimal	586.91	0	4.47e−103	13	0	0
LISWET12	Optimal	574.81	0	6.47e−106	12	0	0
LISWET2	Optimal	699.25	0	4.17e−107	17	0	0
LISWET3	Optimal	299.09	0	1.20e−101	14	0	0
LISWET4	Optimal	309.00	0	2.60e−107	15	0	0
LISWET5	Optimal	312.78	0	4.27e−106	15	0	0
LISWET6	Optimal	316.60	0	3.68e−102	14	0	0
LISWET7	Optimal	1189.62	0	3.11e−108	37	0	0
LISWET8	Inconsistent	659.38	0	1.63e−104	15	0	0
LISWET9	Optimal	671.45	0	2.04e−104	10	0	0
LOTSCHD	Optimal	0.00	0	2.70e−107	6	0	0
MOSARQP1	Optimal	2.14	0	2.01e−115	10	0	0
MOSARQP2	Optimal	0.83	0	1.11e−107	9	0	0
POWELL20	Optimal	127.81	0	2.82e−110	9	0	0
PRIMAL1	Optimal	0.14	0	1.35e−112	7	0	0
PRIMAL2	Optimal	0.27	0	2.38e−103	7	0	0
PRIMAL3	Optimal	0.39	0	1.38e−102	6	0	0
PRIMAL4	Optimal	0.72	0	1.06e−112	7	0	0

Table 8 continued

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
PRIMALC1	Optimal	0.04	0	1.56e−104	6	0	0
PRIMALC2	Optimal	0.01	0	8.59e−107	6	0	0
PRIMALC5	Optimal	0.01	0	1.63e−103	6	0	0
PRIMALC8	Inconsistent	0.15	0	NaN	50	0	0
Q25FV47	Optimal	9.15	0	3.75e−114	9	0	0
QADLITTL	Optimal	0.03	0	2.11e−111	7	0	0
QAFIRO	Error	0.01	0	NaN	9	0	0
QBANDM	Optimal	0.24	0	1.17e−103	6	0	0
QBEACONF	Inconsistent	0.21	0	NaN	50	0	0
QBORE3D	Inconsistent	0.15	0	NaN	50	0	0
QBRANDY	Optimal	0.18	0	4.04e−106	9	0	0
QCAPRI	Optimal	0.27	0	8.03e−116	10	0	0
QE226	Optimal	0.21	0	4.36e−107	8	0	0
QETAMACR	Optimal	1.24	0	3.85e−106	10	0	0
QFFFFF80	Optimal	0.97	0	3.05e−114	9	0	0
QFORPLAN	Optimal	0.23	0	1.88e−113	8	0	0
QGFRDXPN	Optimal	0.41	0	3.34e−104	9	0	0
QGROW15	Optimal	0.32	0	2.68e−107	9	0	0
QGROW22	Optimal	0.58	0	1.77e−107	9	0	0
QGROW7	Optimal	0.10	0	4.66e−109	9	0	0
QISRAEL	Optimal	0.19	0	9.15e−104	9	0	0
QPCBLEND	Optimal	0.03	0	4.29e−114	9	0	0
QPCBOEI1	Optimal	0.50	0	5.70e−114	10	0	0
QPCBOEI2	Optimal	0.12	0	6.52e−105	9	0	0
QPCSTAIR	Optimal	0.95	0	1.04e−109	11	0	0
QPILOTNO	Inconsistent	40.22	0	NaN	50	0	0
QPTEST	Optimal	0.00	0	2.66e−110	6	0	0
QRECIPE	Inconsistent	0.02	0	NaN	50	0	0
QSC205	Optimal	0.27	0	1.27e−107	11	0	0
QSCAGR25	Optimal	0.32	0	1.06e−104	7	0	0
QSCAGR7	Optimal	0.02	0	3.71e−107	8	0	0
QSCFXM1	Optimal	0.16	0	1.79e−112	7	0	0
QSCFXM2	Optimal	0.68	0	5.01e−112	9	0	0
QSCFXM3	Optimal	1.50	0	3.59e−101	8	0	0
QSCORPIO	Optimal	0.16	0	2.31e−101	8	0	0

Table 8 continued

QP Name	Status	Time [s]	Iter. [#]	Tol. [–]	Ref. [#]	Back. [#]	Res. [#]
QSCRS8	Optimal	0.36	0	1.58e−108	13	0	0
QSCSD1	Optimal	0.09	0	7.72e−110	7	0	0
QSCSD6	Optimal	0.23	0	2.69e−111	7	0	0
QSCSD8	Optimal	8.62	0	2.23e−106	10	1	0
QSCTAP1	Optimal	0.19	0	6.15e−106	7	0	0
QSCTAP2	Optimal	0.55	0	5.86e−111	6	0	0
QSCTAP3	Optimal	0.92	0	7.96e−110	6	0	0
QSEBA	Optimal	0.30	0	1.15e−103	9	0	0
QSHARE1B	Optimal	0.04	0	5.84e−112	7	0	0
QSHARE2B	Optimal	0.05	0	1.65e−105	9	0	0
QSHELL	Optimal	0.80	0	3.16e−103	9	0	0
QSHIP04L	Optimal	0.58	0	7.13e−110	9	0	0
QSHIP04S	Optimal	0.43	0	5.88e−109	9	0	0
QSHIP08L	Optimal	2.06	0	2.31e−110	7	0	0
QSHIP08S	Optimal	1.18	0	2.31e−108	9	0	0
QSHIP12L	Optimal	2.67	0	3.71e−113	8	0	0
QSHIP12S	Optimal	2.92	0	2.82e−104	9	0	0
QSIERRA	Error	0.95	0	NaN	6	0	0
QSTAIR	Optimal	0.66	0	4.32e−101	8	0	0
QSTANDAT	Optimal	0.11	0	3.62e−107	7	0	0
S268	Inconsistent	0.00	0	6.35e−112	9	0	0
STADAT1	Optimal	15.69	0	2.28e−107	8	0	0
STADAT2	Optimal	16.69	0	2.33e−110	11	0	0
STADAT3	Optimal	116.94	0	8.80e−104	12	1	0
STCQP1	Optimal	47.31	0	1.91e−107	11	0	0
STCQP2	Optimal	25.51	0	5.65e−102	9	0	0
TAME	Optimal	0.00	0	3.12e−112	6	0	0
UBH1	Optimal	203.99	0	3.28e−111	7	0	0
VALUES	Inconsistent	0.08	0	NaN	50	0	0
YAO	Optimal	23.19	0	1.58e−106	10	0	0
ZECEVIC2	Optimal	0.00	0	2.66e−110	6	0	0

Iter. iterations, *Tol.* tolerance, *Ref.* refinements, *Back.* backstepping, *Res.* resolves

Table 9 Detailed results for inexact and fast solves of medium QP set with parameter set s5 and the three standard qpOASES option sets

QP Name	Refinement Tol. 1e−12			qpOASES with standard settings								
				MPC			Default			Reliable		
	Time [s]	Iter. (Ref.) [#] ([#])	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]
CVXQP1_M	9.53	412 (1)	8.50e−21	6.94	404	2.17e−10	23.92	1705	2.21e−11	169.24	1705	2.34e−11
CVXQP1_S	0.02	36 (0)	2.73e−13	0.02	36	4.05e−12	0.02	139	4.39e−13	0.06	139	3.88e−13
CVXQP2_M	10.50	651 (0)	1.05e−12	9.48	663	5.65e−11	6.84	712	1.50e−12	44.61	713	8.41e−13
CVXQP2_S	0.02	62 (0)	7.47e−14	0.02	60	1.29e−11	0.01	68	9.94e−14	0.01	68	5.82e−14
CVXQP3_M	4.52	231 (2)	8.10e−22	3.86	229	5.40e−09	79.12	3867	3.95e−10	307.04	3869	3.20e−10
CVXQP3_S	0.02	22 (0)	7.18e−12	0.02	24	5.30e−11	0.07	189	4.73e−13	0.05	181	2.50e−13
DPKLO1	0.02	0 (0)	2.67e−14	0.02	0	2.67e−14	0.14	321	5.75e−15	0.31	321	7.32e−15
DUAL1	0.03	26 (1)	5.14e−17	0.02	28	1.09e−11	0.04	75	8.78e−16	0.02	75	7.06e−16
DUAL2	0.04	4 (1)	6.32e−17	0.03	4	3.05e−11	0.05	92	6.06e−16	0.05	92	7.44e−16
DUAL3	0.05	14 (1)	2.53e−17	0.03	14	1.68e−12	0.06	97	1.03e−15	0.06	97	9.27e−16
DUAL4	0.01	13 (0)	1.01e−15	0.01	13	1.51e−11	0.03	62	1.04e−15	0.02	62	8.03e−16
DUALC1	0.08	31 (0)	4.40e−12	0.01	31	2.18e−10	0.01	4	6.20e−13	0.00	4	6.25e−13
DUALC2	0.14	28 (1)	1.91e−21	0.01	30	2.17e−09	0.00	5	2.33e−13	0.00	5	2.73e−13
DUALC5	0.10	3 (0)	3.32e−13	0.01	3	1.30e−09	0.01	5	2.68e−13	0.00	5	8.70e−14
DUALC8	0.60	13 (1)	6.10e−19	0.02	11	8.95e−10	0.01	6	5.15e−11	0.01	0	NaN
GENHS28	0.00	0 (0)	3.47e−16	0.00	0	3.47e−16	0.00	14	6.80e−16	0.00	14	5.13e−16
GOULDQP2	6.53	585 (2)	9.00e−21	3.77	573	7.77e−14	15.44	2409	1.13e−09	247.95	2409	1.13e−09
GOULDQP3	1.68	176 (0)	7.52e−15	1.54	176	5.94e−13	2.98	740	8.40e−15	17.52	740	8.16e−15
HS118	0.00	24 (0)	4.71e−15	0.00	24	6.76e−13	0.00	27	6.92e−15	0.00	27	5.56e−15
HS21	0.00	3 (0)	8.80e−16	0.00	3	1.28e−17	0.00	1	5.55e−17	0.00	1	5.55e−17
HS268	0.00	0 (1)	1.69e−16	0.00	0	2.62e−12	0.00	11	8.60e−07	0.00	12	7.87e−13

Table 9 continued

QP Name	Refinement Tol. 1e−12			qpOASES with standard settings								
				MPC			Default			Reliable		
	Time [s]	Iter. (Ref.) [#] ([#])	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]
HS35	0.00	1 (0)	3.61e−16	0.00	1	2.22e−16	0.00	4	7.77e−16	0.00	4	3.06e−16
HS35MOD	0.00	0 (0)	1.01e−15	0.00	0	9.98e−16	0.00	4	1.66e−16	0.00	4	1.66e−16
HS51	0.00	0 (0)	2.54e−16	0.00	0	2.54e−16	0.00	5	8.16e−15	0.00	5	8.16e−15
HS52	0.00	0 (0)	1.22e−15	0.00	0	1.22e−15	0.00	10	2.29e−15	0.00	10	1.13e−15
HS53	0.00	0 (0)	3.89e−16	0.00	0	3.89e−16	0.00	9	1.00e−15	0.00	9	8.30e−16
HS76	0.00	4 (0)	3.16e−17	0.00	4	2.78e−16	0.00	4	6.36e−16	0.00	4	6.36e−16
KSIP	24.26	1080 (1)	1.63e−18	0.15	1088	3.61e−15	0.21	1019	1.89e−16	0.20	1019	3.68e−17
LOTSCHD	0.00	5 (0)	8.82e−15	0.00	5	5.30e−13	0.00	18	1.82e−14	0.00	18	2.27e−14
MOSARQP2	18.25	332 (0)	1.18e−15	2.54	332	6.33e−13	8.99	1012	1.83e−15	317.07	1012	1.32e−15
PRIMAL1	0.38	77 (0)	1.92e−16	0.17	75	1.76e−11	0.50	399	2.29e−16	6.14	399	1.05e−15
PRIMAL2	3.49	94 (0)	8.16e−16	0.52	96	4.02e−11	3.04	742	1.45e−15	85.84	742	1.68e−15
PRIMAL3	2.95	117 (0)	1.21e−15	0.69	101	4.77e−11	4.86	841	1.48e−15	134.65	841	1.83e−16
PRIMALC1	0.17	234 (1)	2.69e−22	0.10	222	2.47e−09	0.02	27	6.50e−13	0.01	27	1.01e−12
PRIMALC2	0.16	237 (1)	1.01e−22	0.11	237	6.00e−13	0.00	10	8.05e−16	0.00	10	1.66e−13
PRIMALC5	0.27	288 (1)	7.11e−23	0.17	292	3.15e−10	0.03	23	2.50e−14	0.02	23	2.58e−14
PRIMALC8	2.76	515 (1)	3.87e−17	0.70	519	2.34e−09	0.07	25	1.91e−14	0.05	26	4.08e−12
QADLITTL	0.05	234 (1)	1.92e−17	0.03	189	5.93e−09	0.02	132	3.31e−13	0.02	124	1.39e−10

Table 9 continued

QP Name	Refinement Tol. 1e−12			qpOASES with standard settings								
				MPC			Default			Reliable		
	Time [s]	Iter. (Ref.) [#] ([#])	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]
QAFIRO	0.00	30 (0)	5.49e−15	0.00	29	8.37e−11	0.00	16	3.48e−15	0.00	16	3.48e−15
QBANDM	2.85	1164 (0)	1.54e−13	1.59	870	1.61e−08	5.43	1512	2.23e−13	7.69	1512	9.62e−14
QBEACONF	0.43	304 (2)	1.18e−18	0.24	302	2.44e−08	0.09	133	1.30e−11	0.09	133	1.31e−11
QBORE3D	0.78	522 (1)	5.27e−13	0.25	155	NaN	0.30	221	4.61e−13	0.31	221	2.76e−11
QBRANDY	0.56	444 (0)	4.69e−12	0.28	414	2.53e−08	0.85	854	2.43e−13	1.16	854	3.13e−13
QCAPRI	2.69	1050 (1)	3.62e−19	1.06	1008	9.68e−08	0.64	457	4.83e−10	0.91	457	4.34e−10
QE226	3.24	1393 (1)	1.07e−13	0.95	1431	2.59e−08	1.17	825	2.36e−14	2.07	813	3.78e−14
QETAMACR	4.12	558 (2)	6.05e−16	2.15	493	2.14e−09	7.86	1354	2.30e−07	23.13	1354	2.30e−07
QFFFFF80	19.05	1008 (1)	2.26e−13	7.76	1005	9.90e−08	20.85	2293	1.59e−11	73.22	1722	NaN
QFORPLAN	2.21	1180 (1)	7.31e−22	1.53	1930	2.39e+06	1.28	796	8.03e−10	1.81	804	7.13e−10
QGROW15	4.07	629 (1)	8.55e−18	2.66	531	1.04e−07	3.29	600	4.28e−09	4.17	589	6.40e−09
QGROW22	15.30	934 (2)	1.52e−20	7.62	621	1.14e−07	10.71	888	3.69e−07	14.64	881	4.97e−08
QGROW7	0.45	296 (1)	1.56e−19	0.28	266	5.07e−08	0.42	340	1.88e−09	0.45	298	3.22e−09
QISRAEL	0.41	290 (0)	3.11e−12	0.11	360	3.99e−08	0.08	258	4.81e−12	0.12	258	5.03e−12
QPCBLEND	0.03	66 (1)	2.25e−16	0.01	64	8.38e−14	0.02	176	7.56e−16	0.02	176	9.67e−16
QPCBOEI1	3.18	435 (2)	2.12e−13	1.08	441	2.15e−09	1.32	652	2.74e−11	7.68	652	3.20e−11
QPCBOEI2	0.30	175 (0)	6.07e−11	0.07	177	7.37e−10	0.07	224	4.21e−10	0.16	224	3.00e−10

Table 9 continued

QP Name	Refinement Tol. 1e−12			qpOASES with standard settings								
				MPC			Default			Reliable		
	Time [s]	Iter. (Ref.) [#] ([#])	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]	Time [s]	Iter. [#]	Tol. [–]
QPCSTAIR	1.32	257 (0)	1.58e−11	0.54	239	9.26e−11	2.29	908	8.77e−12	3.67	908	8.06e−12
QPTEST	0.00	1 (0)	7.40e−16	0.00	1	1.22e−15	0.00	2	3.89e−16	0.00	2	3.89e−16
QRECIPE	0.07	48 (0)	5.90e−15	0.02	42	1.04e−10	0.02	88	1.64e−14	0.02	88	1.64e−14
QSC205	0.29	93 (1)	4.22e−18	0.04	43	7.63e−08	0.08	215	3.84e−16	0.08	215	4.35e−16
QSCAGR25	7.38	1077 (2)	1.77e−18	36.34	0	NaN	6.12	1327	1.09e−11	8.88	1299	1.43e−11
QSCAGR7	0.19	329 (0)	3.88e−12	0.06	349	3.18e−10	0.22	418	4.58e−12	0.22	418	5.03e−12
QSCFXM1	4.18	799 (2)	1.25e−15	1.73	988	1.32e−07	1.69	580	9.14e−12	2.69	612	1.48e−11
QSCFXM2	42.74	1890 (2)	3.82e−18	16.53	2189	1.62e−07	17.45	1454	6.84e−11	26.22	1370	3.92e−11
QSCORPIO	0.87	233 (0)	5.22e−14	0.12	0	NaN	0.87	470	3.20e−11	1.19	469	1.06e−11
QSCSD1	8.81	2054 (1)	3.44e−22	6.51	1075	8.77e−11	0.64	204	4.77e−13	0.57	153	1.38e−10
QSCTAP1	6.77	1374 (1)	7.45e−15	30.15	0	NaN	1.00	500	9.20e−12	1.44	503	1.54e−08
QSHARE1B	0.59	782 (2)	1.25e−16	0.18	468	2.49e−08	0.34	517	1.21e−09	0.51	497	3.70e−11
QSHARE2B	0.12	359 (1)	2.00e−13	0.02	355	5.33e−10	0.06	207	1.35e−12	0.05	196	1.25e−12
QSTAIR	3.18	792 (0)	6.37e−12	1.28	784	1.92e−08	1.68	740	4.39e−12	2.67	740	4.41e−12
S268	0.00	0 (1)	1.69e−16	0.00	0	2.62e−12	0.00	11	8.60e−07	0.00	12	7.87e−13
TAME	0.00	0 (0)	1.11e−16	0.00	0	1.11e−16	0.00	2	1.95e−16	0.00	2	1.95e−16
VALUES	0.10	142 (0)	1.46e−12	0.04	0	NaN	0.05	142	4.29e−16	0.04	142	2.45e−15
ZECEVIC2	0.00	3 (0)	1.25e−16	0.00	3	1.15e−12	0.00	2	2.22e−16	0.00	2	2.22e−16

Iter. iterations, *Tol.* tolerance

References

1. Bennett, K.P., Campbell, C.: Support vector machines: hype or hallelujah? ACM SIGKDD Explor. Newsl. **2**(2), 1–13 (2000)
2. CPLEX, IBM ILOG: 12.7 user's manual. https://www.ibm.com (2016). Accessed 25 Oct 2019
3. Dostál, Z.: Optimal Quadratic Programming Algorithms: With Applications to Variational Inequalities, 1st edn. Springer Publishing Company, Incorporated, Berlin (2009)
4. Ferreau, H.J., Kirches, C., Potschka, A., Bock, H.G., Diehl, M.: qpOASES: a parametric active-set algorithm for quadratic programming. Math. Program. Comput. **6**(4), 327–363 (2014)
5. Fletcher, R., Leyffer, S.: User manual for filterSQP. Numerical Analysis Report NA/181, Department of Mathematics, University of Dundee, Dundee, Scotland (1998)
6. Gärtner, B., Schönherr, S.: An efficient, exact, and generic quadratic programming solver for geometric optimization. In: Proceedings of the Sixteenth Annual Symposium on Computational Geometry, pp. 110–118. ACM (2000)
7. Gill, P.E., Murray, W., Saunders, M.A.: User's guide for QPOPT 1.0: A Fortran package for quadratic programming. Technical report (1995)
8. Gleixner, A.M.: Exact and fast algorithms for mixed-integer nonlinear programming. Ph.D. thesis, Technische Universität Berlin (2015)
9. Gleixner, A.M., Steffy, D.E., Wolter, K.: Improving the accuracy of linear programming solvers with iterative refinement. In: ISSAC 2012. Proceedings of the 37th International Symposium on Symbolic and Algebraic Computation, pp. 187–194. ACM (2012)
10. Gleixner, A.M., Steffy, D.E., Wolter, K.: Iterative refinement for linear programming. INFORMS J. Comput. **28**(3), 449–464 (2016). https://doi.org/10.1287/ijoc.2016.0692
11. Goldberg, D.: What every computer scientist should know about floating-point arithmetic. ACM Comput. Surv. **23**(1), 5–48 (1991). https://doi.org/10.1145/103162.103163
12. Goldfarb, D., Idnani, A.: Dual and primal-dual methods for solving strictly convex quadratic programs. In: Hennart J.P. (eds.) Numerical Analysis. Lecture Notes in Mathematics, vol 909. Springer, Heidelberg (1982)
13. Gould, N.I., Hribar, M.E., Nocedal, J.: On the solution of equality constrained quadratic programming problems arising in optimization. SIAM J. Sci. Comput. **23**(4), 1376–1395 (2001)
14. Granlund, T., the GMP Development Team: GNU MP. The GNU Multiple Precision Arithmetic Library. Edition 6.1.0. https://gmplib.org/. Accessed 25 Jan 2019
15. Gurobi Optimization, I.: Gurobi optimizer reference manual. Technical Report Version 7.5 (2017). http://www.gurobi.com. Accessed 25 Oct 2017
16. Hladık, M.: Interval linear programming: a survey. In: Mann, Z.Á. (ed.) Linear programming–new frontiers in theory and applications, pp. 85–120. Nova Science Publishers, New York (2012)
17. IEEE, New York, NY, USA: IEEE Std 754-2008, Standard for Floating-Point Arithmetic (2008). https://doi.org/10.1109/IEEESTD.2008.4610935
18. Johnson, T.C., Kirches, C., Wächter, A.: An active-set method for quadratic programming based on sequential hot-starts. SIAM J. Optim. **25**(2), 967–994 (2015)
19. Maros, I., Mészáros, C.: A repository of convex quadratic programming problems. Optim. Methods Softw. **11**(1–4), 671–681 (1999)
20. Quirynen, R., Gros, S., Diehl, M.: Inexact newton-type optimization with iterated sensitivities. SIAM J. Optim. **28**(1), 74–95 (2018)
21. Weber, T., Gleixner, A.: QPRefinement (2019). https://doi.org/10.5281/zenodo.2532184. https://github.com/TobiasWeber/QPRefinement
22. Wilkinson, J.H.: Rounding errors in algebraic processes. In: IFIP Congress, pp. 44–53 (1959)
23. Wolfe, P.: The simplex method for quadratic programming. Econom. J. Econom. Soc. **27**, 382–398 (1959)
24. Wunderling, R.: Paralleler und Objektorientierter Simplex-Algorithmus. Ph.D. thesis, Technische Universität Berlin (1996)