Math. Prog. Comp. (2013) 5:113–142
DOI 10.1007/s12532-012-0050-3

FULL LENGTH PAPER

Trajectory-following methods for large-scale degenerate convex quadratic programming

Nicholas I. M. Gould · Dominique Orban · Daniel P. Robinson

Received: 4 September 2011 / Accepted: 12 December 2012 / Published online: 16 February 2013

Abstract We consider a class of infeasible, path-following methods for convex quadratic programming. Our methods are designed to be effective for solving both nondegerate and degenerate problems, where degeneracy is understood to mean the failure of strict complementarity at a solution. Global convergence and a polynomial bound on the number of iterations required is given. An implementation, CQP, is available as part of GALAHAD. We illustrate the advantages of our approach on the CUTEr and Maros–Meszaros test sets.

Keywords Convex quadratic programming · Path-following methods · Degenerate problems · Software

Mathematics Subject Classification (2000) 65K05 · 90C20 · 90C25 · 90C51

Electronic supplementary material The online version of this article (doi:10.1007/s12532-012-0050-3) contains supplementary material, which is available to authorized users.

N. I. M. Gould (✉)
Scientific Computing Department, Rutherford Appleton Laboratory,
Chilton, Oxfordshire, OX11 0QX, UK
e-mail: nick.gould@stfc.ac.uk

D. Orban
GERAD and Mathematics and Industrial Engineering Department, École Polytechnique de Montréal,
C. P. 6079, Succ. Centre Ville, Montreal, QC H3C 3A7, Canada
e-mail: dominique.orban@gerad.ca

D. P. Robinson
Department of Applied Mathematics and Statistics, Johns Hopkins University,
100 Whitehead Hall, 3400 N. Charles Street, Baltimore, MD 21218, USA
e-mail: daniel.p.robinson@jhu.edu

1 Introduction

1.1 The problem, optimality conditions and central path

We consider an infeasible interior-point method for solving the general convex quadratic program

$$\textbf{CQP}: \underset{x\in\mathbb{R}^n}{\text{minimize}} \ q(x)=\tfrac{1}{2}x^T Hx+g^T x \quad \text{subject to} \quad c^{\mathrm{L}} \le Ax \le c^{\mathrm{U}} \quad \text{and} \quad x^{\mathrm{L}} \le x \le x^{\mathrm{U}}, \tag{1.1}$$

where $H \in \mathbb{R}^{n\times n}$ is a positive semi-definite matrix, $A \in \mathbb{R}^{m\times n}$ is assumed to have full rank, and any of the components of the vectors $c^{\mathrm{L}} \le c^{\mathrm{U}} \in \mathbb{R}^m$ and $x^L \le x^U \in \mathbb{R}^n$ may be infinite. The full-rank assumption on A is not significant since in practice any dependencies may be removed during preprocessing; we do this in our implementation. For ease of exposition, we focus on the special case

$$\textbf{QP}: \underset{x\in\mathbb{R}^n}{\text{minimize}} \ q(x) = \tfrac{1}{2}x^T Hx + g^T x \quad \text{subject to} \quad Ax = b \quad \text{and} \quad x \ge 0$$

that involves equality constraints and non-negativity bounds, but will return to the general case when providing details of our implementation.

Any local solution x_* to QP necessarily satisfies the first-order optimality (KKT) conditions

$$Hx_* + g = A^T y_* + z_*, \quad Ax_* = b, \quad \text{and} \quad x_* \cdot z_* = 0, \tag{1.2}$$

involving the primal variables $x_* \ge 0$, dual variables $z_* \ge 0$ and Lagrange multipliers y_*, where $\cdot$ denotes the component-wise product; the three conditions in (1.2) are known as dual feasibility, primal feasibility, and complementary slackness or complementarity, respectively. For brevity we define $v = (x, y, z)$, $v_* = (x_*, y_*, z_*)$, etc.

We say that v is *feasible* for problem QP if it belongs to the set

$$\mathcal{F} \overset{\text{def}}{=} \{v \in \mathbb{R}^{2n+m} : Hx+g-A^T y-z=0, \quad Ax=b, \quad x\ge 0, \quad \text{and} \quad z\ge 0\} \tag{1.3}$$

and *strictly feasible* if it belongs to the set

$$\mathcal{F}_0 \overset{\text{def}}{=} \{v \in \mathbb{R}^{2n+m} : Hx+g-A^T y-z=0, \quad Ax=b, \quad x>0, \quad \text{and} \quad z>0\}. \tag{1.4}$$

Thus, a solution (x_*, y_*, z_*) of QP satisfies $(x_*, y_*, z_*) \in \mathcal{F}$ and $x_* \cdot z_* = 0$. Problem QP is *degenerate* if for every solution there is at least one i for which $[x_*]_i = [z_*]_i = 0$, otherwise it is *non-degenerate*.

The vast majority of feasible interior-point methods approximate a solution v_* by tracing the so-called central path. The *central path* $v(\mu) = \big(x(\mu), y(\mu), z(\mu)\big)$ associated with QP satisfies

$$\begin{aligned} Hx(\mu) - A^T y(\mu) - z(\mu) &= -g, \\ Ax(\mu) &= b, \quad \text{and} \\ x(\mu) \cdot z(\mu) &= \mu e, \end{aligned} \tag{1.5}$$

where the parameter $\mu > 0$ and e is a vector of ones. These equations may be interpreted as perturbed optimality conditions, c.f. (1.2). If $\big(x(\mu), z(\mu)\big) > 0$ for all $\mu >$ and the $\lim_{\mu\to 0} v(\mu)$ exists, then the limit point is necessarily a solution to QP.

1.2 The influence of degeneracy: two examples

Example 1—a non-degenerate QP
Consider the simple quadratic program

$$\underset{x}{\text{minimize}} \ \tfrac{1}{2}x^2 \quad \text{subject to} \quad x \ge 2$$

that has the unique solution $x_* = z_* = 2$. The central path (1.5) for the problem is given by $x(\mu) = z(\mu) = 1 + \sqrt{1+\mu}$. A typical interior-point algorithm—in this case, specifically CQP from GALAHAD [13] with default options—starting from $x = 3$ and $z = 2$ produces the output

```
Iter p-feas  d-feas   com-slk obj     step     target
0    0.0E+00 1.0E+00  2.0E+00 4.5E+00 -        2.0E-02
1    0.0E+00 0.0E+00  1.3E-02 2.0E+00 1.0E+00  1.3E-04
2    0.0E+00 4.4E-16  1.8E-04 2.0E+00 1.0E+00  1.8E-06
3    0.0E+00 0.0E+00  1.8E-06 2.0E+00 1.0E+00  2.4E-09
4    0.0E+00 4.4E-16  2.4E-09 2.0E+00 1.0E+00  1.2E-13
5    0.0E+00 4.4E-16  1.2E-13 2.0E+00 1.0E+00  3.9E-20
```

where Iter is the iteration number, p-feas, d-feas and comp-slk are the violations of primal and dual feasibility and the complementary slackness, obj is the objective function value, step is the stepsize and target is the target value of the parameter μ for the next iteration. This illustrates generic behaviour for a good interior-point solver. The solution obtained is correct to twelve decimal digits. In anticipation of what follows, note that $x(\mu)$ is analytic for $\mu \ge 0$.

Example 2—a degenerate QP
Now consider the variant

$$\underset{x}{\text{minimize}} \ \tfrac{1}{2}x^2 \quad \text{subject to} \quad x \ge 0$$

whose solution is $x_* = z_* = 0$. For this problem, the central path is $x(\mu) = z(\mu) = \sqrt{\mu}$, and the same interior-point solver now reports

```
Iter p-feas  d-feas  com-slk  obj     step    target
0    0.0E+00 1.0E+00 2.0E+00  5.0E-01 -       2.0E-02
1    0.0E+00 0.0E+00 4.5E-01  2.3E-01 1.0E+00 4.5E-03
2    0.0E+00 0.0E+00 1.2E-01  5.8E-02 1.0E+00 1.2E-03
```

```
3     0.0E+00 0.0E+00 2.9E-02  1.5E-02 1.0E+00 2.9E-04
4     0.0E+00 0.0E+00 7.5E-03  3.8E-03 1.0E+00 7.5E-05
5     0.0E+00 0.0E+00 1.9E-03  9.6E-04 1.0E+00 1.9E-05
6     0.0E+00 3.5E-18 4.9E-04  2.4E-04 1.0E+00 4.9E-06
..    ....... ....... .......  ....... ....... .......
18    0.0E+00 0.0E+00 3.1E-11  1.6E-11 1.0E+00 1.7E-16
19    0.0E+00 0.0E+00 7.8E-12  3.9E-12 1.0E+00 2.2E-17
20    0.0E+00 6.4E-22 1.9E-12  9.7E-13 1.0E+00 2.7E-18
21    0.0E+00 3.2E-22 4.8E-13  2.4E-13 1.0E+00 3.4E-19
```

when started from $x = 1$ and $z = 2$. Why is the convergence so slow? Simply, the Newton iteration at the heart of the solver is predicated on the solution trajectory being analytic while clearly $x(\mu)$ fails to be analytic at the limit $\mu = 0$. The QP is degenerate, since the distance of x_* from its lower bound and z_* are both zero. Behaviour such as the above is generic for Taylor-series-based methods for degenerate quadratic programs that try to trace $v(\mu)$. Worse, on termination the approximations to x_* and z_* are only correct to six decimal digits.

Is following the central path to blame? Actually, no, it is simply that the parameterization used in (1.5) is inappropriate.

Example 2 again
Again, consider Example 2, but now simply transform the parameter so that $\mu = \rho^2$ and note that in this case $x(\rho) = z(\rho) = \rho$, which is a perfectly analytic function of ρ. Applying a Taylor-series-based method to this reformulation within CQP yields

```
Iter p-feas  d-feas  com-slk  obj     step    target
0    0.0E+00 1.0E+00 2.0E+00  5.0E-01 -       2.0E-02
1    0.0E+00 1.1E-16 5.6E-01  2.8E-01 1.0E+00 5.6E-03
2    0.0E+00 1.1E-16 5.6E-05  2.8E-05 1.0E+00 4.2E-07
3    0.0E+00 1.1E-16 3.1E-09  1.5E-09 1.0E+00 1.7E-13
4    0.0E+00 1.1E-16 9.6E-18  4.8E-18 1.0E+00 3.0E-26
```

which mimics the behaviour in the non-degenerate case and produces errors that are comparable with those for Example 1.

Our intention in this paper is to describe the ideas behind the GALAHAD QP solver CQP that has been designed, with these examples in mind, to produce fast accurate solutions for both degenerate and non-degenerate problems.

1.3 Perspective

Extensive research on interior-point methods for solving QP has resulted in a large number of algorithms and theorems. In fact, every algorithm designed for the linear complementarity problem (LCP) [21], horizontal linear complementarity problem (hLCP) [2,32,51,52], monotone linear complementarity problem (mLCP) [20,33,34,43,45], geometrical linear complementarity problem (gLCP) [30], and sufficient linear complementarity problem (sLCP) [22,23,35,36,39–42,44], may be used to solve QP since convex quadratic programming is contained within these problem classes (see [1,48] for descriptions and equivalences between different formulations). Moreover, advances in linear programming [4,11,28,29,31,50] are pertinent since they often lead to improved algorithms for solving QP; a good example is the well-known predictor-corrector scheme that Mehrotra [29] popularized.

The references above—although plentiful—represent a small subset of the total research in the area. We believe, however, that every proposed method can be categorised by its (1) restrictions on an initial iterate; (2) step generation; (3) criteria for step acceptance; (4) complexity results; (5) global convergence guarantees; (6) rate of local convergence; (7) *theoretical* performance on nondegenerate *and* degenerate problems; (8) number of factorizations per iteration; and (9) documented *practical* performance. We now discuss these in detail.

Feasible interior-point methods [20,22,23,33,34,45] require a strictly feasible starting point v, while *infeasible* interior-point methods [2,4,11,29,30,35,36,39,41,44,51,52] only need $(x, z) > 0$. Generally, infeasible interior-point methods are preferred since a starting point may be obtained without excessive computation. In fact, the widely used solvers BPMPD, Cplex, HOPDM, Mosek, OOPS, OOQP, PCx, and Xpress are all of the latter type.

The methods we consider are iterative. A sequence of iterates is generated by approximately following an "ideal" arc that emanates from the current point and leads either directly to a solution of QP or to some suitable point on the central path. We shall return to this in Sect. 2.

The step acceptance criteria has a strong influence on the ultimate performance of any algorithm. Every criteria includes a condition that ensures that the iterates remain strictly positive and (usually) in the vicinity of the so-called (weighted) central path. There are four common approaches. The so-called fraction-to-the-boundary rule forces the updated iterate to be a distance from the boundary of $(x, z) \geq 0$ that is at least some fraction of the distance from the current point to the boundary. Algorithms based on this approach [4,11,29] typically work very well in practice, but usually lack a proof of global convergence. The other strategies force the iterates to stay "centered" by requiring either

$$\left\|c^{\text{CS}}(v)-e\right\|_2 \leq (1-\gamma) \quad \text{or equivalently} \quad \|r^{\text{CS}}(v)-\mu(v)e\|_2 \leq (1-\gamma)\mu(v), \tag{1.6}$$

or

$$\left\|\max\left[0, e - c^{\text{CS}}(v)\right]\right\|_\infty \leq (1-\gamma) \quad \text{or equivalently} \quad r^{\text{CS}}(v) \geq \gamma\mu(v)e, \tag{1.7}$$

or

$$\frac{1}{\gamma}\mu(v)e \geq r^{\text{CS}}(v) \geq \gamma\mu(v)e \tag{1.8}$$

for some $\gamma \in (0, 1)$, where

$$r^{\text{CS}}(v) \overset{\text{def}}{=} x \cdot z, \quad \mu(v) \overset{\text{def}}{=} \frac{x^T z}{n}, \quad \text{and} \quad c^{\text{CS}}(v) \overset{\text{def}}{=} \frac{r^{\text{CS}}(v)}{\mu(v)} \tag{1.9}$$

are the *pairwise complementarity slackness function*, the *mean complementarity function*, and the *centering function* respectively. Condition (1.6) is the most restrictive, but still used by many authors [20,30,41,44,45]. For a given search arc, condition (1.6) often dictates significant backtracking, and this smothering of the step results in good complexity at the expense of practical performance. The so-called "wide" neighborhood described by (1.7) is not overly restrictive, but still guarantees that each complementarity pair $x_i z_i$ is never too much smaller than the mean complementarity $\mu(v)$. Consequently, algorithms that use this condition (see [2,11,22,23,33,34,36,51,52]) may work well in practice, but their complexity results usually suffer. This observation highlights the disconnection between theory and practice since algorithms with the best complexity often perform poorly in practice. Finally, condition (1.8) ensures that each complementarity pair is never too much smaller *or* larger than the mean complementarity. Colombo and Gondzio [4] observed that this "symmetric" neighborhood often resolves poor performance resulting from complementarity pairs that are much larger than the mean; the discrepancy in size leads to poorly scaled systems [4] that are solved during each iteration.

Some of the most efficient methods [4,11,29] are based on clever heuristics and do not have a proof of global convergence; this partly explains why these methods occasionally fail. In this paper we introduce an algorithm that is highly efficient and globally convergent.

Many algorithms in the literature generate iterates that are superlinearly convergent assuming that problem QP is nondegenerate [20,52]. Unfortunately, our experience leads us to believe that degenerate problems routinely arise in practice. Thus, it is vital that algorithms are capable of solving degenerate *and* nondegenerate problems efficiently; this has been accomplished (theoretically) by [22,23,30,33–36,39,41,44,45].

We should not judge the numerical performance of a method only by the average number of iterations, but rather by the average number of iterations *and* the computation needed per iteration. For problem QP the main computational expense can often be measured by the number of factorizations. Consequently, algorithms that require a single factorization [2,4,11,29,34–36,39,51,52] per iteration are more attractive than those that require multiple ones [20,22,23,30,33,41,44,45].

We believe that insufficient numerical evidence of the above algorithms is a gross oversight. From the above articles, numerical testing was performed only in [4,11,29], which are all "predictor-corrector" type methods. We believe this has lead the optimization community to accept that other viable algorithms do not exist. In this paper we challenge conventional wisdom by formulating an algorithm that is *not* of the predictor-corrector type, and then verify numerically that it is highly efficient.

Based on the previous discussion, we believe that an ideal interior-point algorithm for solving QP should possess the following properties.

P1. An initial starting point is easily and cheaply obtained.
P2. The search arc is both cheap to obtain and effective.
P3. The conditions for step acceptance are relatively weak.
P4. The iterates converge at a linear rate globally.
P5. The iterates converge at a superlinear rate locally.
P6. The algorithm has polynomial complexity.

P7. The algorithm is capable of obtaining highly accurate solutions (if desired) and performs well on nondegenerate *and* degenerate problems.

We are not aware of any method that satisfies all of these properties. Property P1 avoids excessive computation and the necessity of a "two-phase" method. Properties P2 and P3 are important since well-chosen search arcs and weaker step acceptance criteria often allows longer trial steps to be accepted, which generally results in superior performance. Properties P4–P6 are clearly desirable properties of any algorithm, but P6 should not be obtained by sacrificing P7. Property P7 is crucial since degenerate problems frequently arise in practice and interior-point methods typically do not perform well; usually, the iterates converge slowly and computing highly accurate solutions is not possible. This is problematic in at least two situations. First, a user may be interested in obtaining a highly accurate solution to a "one-off" convex quadratic program; this is not possible with traditional interior-point schemes. Second, the quadratic program may be an auxiliary subproblem for a nonlinear programming method, e.g., sequential quadratic programming (SQP). These methods often need an accurate solution to QP to ensure descent on a so-called merit function and to accurately predict the set of active constraints of a solution to the original optimization problem. This is true of our evolving SQP algorithm S2QP [14,15] and motivates this research.

We formulate an algorithm that satisfies P1–P7 by building on the previous work of others. The paper by Zhang [51] forms the foundation of our method since it already satisfies most of the desired properties. His algorithm is a first-order infeasible-interior-point strategy designed to solve sufficient linear complementarity problems (sLCP). It is known that this class is equivalent to the P_* class that includes positive-semi-definite matrices [46], and that horizontal and mixed LCPs—which includes QP—may be mapped to each other while preserving P_*-itivity [1]. In particular, the class of sufficient linear complementarity problems includes convex QP. During each iteration, Zhang computes a so-called centered and damped Newton step, which is the sum of the Newton step for (1.2) and a step designed to stay sufficiently far from the boundary of $(x, z) \geq 0$; his method has polynomial complexity, linear global convergence, and uses the wide neighborhood (1.7). He used less than ideal conditions for a starting point, but this was circumvented by Billups and Ferris [2]; they proved that Zhang's results held for any starting point satisfying $(x, z) > 0$. Zhang's algorithm currently forms the basis of the linear and separable quadratic programming solver LSQP, which is part of the GALAHAD [13] optimization suite. LSQP performs well on nondegenerate problems and provides additional incentive to use his algorithm as the "work horse" of our method. We note that Zhang and Zhang [52] proved better complexity bounds for a second-order variant of Zhang's method [51]. In summary, a second-order variant of Zhang's method satisfies every desired property except for P5 and P7, i.e., it is linearly convergent and does not perform well on degenerate problems.

If problem QP is degenerate, then, as we saw in Sect. 1.2, the central path $v(\mu)$ defined by (1.5) is not analytic in μ [43, Thm.2] and has a singularity at $\mu = 0$. This is particularly troublesome since interior-point methods must drive μ towards zero to obtain an approximate solution of problem QP. To resolve this issue, we rely on the relatively unknown work by Stoer and Wechs [42,43]. For degenerate QP, they show that there exists a simple nonlinear re-parameterization of the central path such that the resulting function has an analytic extension to zero. Therefore, degeneracy is not a problem if one views the central path defined by the "correct" parameterization.

Finally, to accelerate local convergence we compute additional trial iterates derived from perturbed trajectories. The trajectories emanate from the current point and lead to a "desirable" location; they have been studied by Potra and Stoer [36], Stoer and Wechs [41], Stoer et al. [44], and Sun and Zhao [53].

Our work contains three key contributions. First, we believe that our algorithm is the first to satisfy properties P1–P7. Second, we combine ideas from Zhang [51], Stoer and Wechs [42,43], Stoer, Wechs, and Mizuno [44], and Sun and Zhao [53] into a single cohesive framework. This framework is quite general and should allow us to seamlessly incorporate any future advances in the field. Finally, we provide details of our implementation and extensive numerical testing; we believe this is the first numerical verification of the ideas presented in [36,41,43,44,53].

The paper is organized as follows. In Sect. 2 we describe and formally state our general algorithm. Our method allows for the computation of additional trial steps that are designed to accelerate convergence; these steps are explored in Sect. 4. We provide convergence results in Sect. 3, a detailed description of our implementation in Sect. 5, numerical testing in Sect. 6, and concluding remarks in Sect. 7.

2 The general algorithm

In this section we describe a rather general algorithm. Using (1.9), we define the *primal-dual infeasibility function*

$$r^{\text{PD}}(v) \equiv \begin{pmatrix} r^{\text{D}}(v) \\ r^{\text{P}}(v) \end{pmatrix} \overset{\text{def}}{=} \begin{pmatrix} Hx + g - A^T y - z \\ Ax - b \end{pmatrix}, \tag{2.1}$$

the *KKT-violation function*

$$r(v) = \begin{pmatrix} r^{\text{PD}}(v) \\ r^{\text{CS}}(v) \end{pmatrix} \tag{2.2}$$

and the *feasibility function*

$$c^{\text{F}}(v) = \mu(v) - \nu \| r^{\text{PD}}(v) \|, \tag{2.3}$$

where $\nu \in (0, 1)$ is a constant. We consider a point (x, y, z) to be *sufficiently centered* if

$$(x, z) > 0 \quad \text{and} \quad \gamma_{\text{L}} e \leq c^{\text{CS}}(v) \leq \gamma_{\text{U}} e, \tag{2.4}$$

for some $0 < \gamma_{\text{L}} < 1 < \gamma_{\text{U}}$, and *sufficiently feasible* if

$$c^{\text{F}}(v) \geq 0. \tag{2.5}$$

Note that condition (2.4) is similar to (1.8) and ensures that individual complementarity components do not deviate too much from the mean, and requirement (2.5) ensures that primal and dual feasibility are achieved when (or before) complementarity vanishes. It follows that if a sequence $\{v_k\}_{k\geq 0}$ satisfies (2.4), (2.5), and $\lim_{k\to\infty} x_k^T z_k = 0$, then any limit point of the sequence is a solution to problem QP. It is natural, therefore, to use the merit function

$$\phi(v;\tau) = x^T z + \tau \|r^{\text{PD}}(v)\| \tag{2.6}$$

to monitor progress of the iterates, where $\tau \geq 0$ is some weighting parameter. Note that this merit function implicitly assumes that the points x and z are positive.

We may now describe a generic iteration of the algorithm. Let v_k be the k-th iterate. We presume that v_k lies within the neighbourhood defined by (2.4) and (2.5) of the central path, but do not insist that it is primal-dual feasible, i.e., $r^{\text{PD}}(v_k)$ may be nonzero. We define an *idealised* (perhaps, infeasible) *solution trajectory* $v_k(\alpha)$ of the scalar $\alpha \in [0, 1]$ via the homotopy

$$r(v_k(\alpha)) = t_k(\alpha), \tag{2.7}$$

where the *target KKT-residual trajectory* (or *residual trajectory* for short) $t_k(\alpha)$ satisfies

$$t_k(0) = r(v_k) \quad \text{and} \quad t_k(1) = t_k$$

for some desired *target residual* t_k for iteration k—the former ensures that

$$v_k(0) = v_k \tag{2.8}$$

so long as the Jacobian

$$\begin{pmatrix} H & -A^T & -I \\ A & 0 & 0 \\ Z_k & 0 & X_k \end{pmatrix}$$

of r at v_k is nonsingular; here and hereafter X_k and Z_k are diagonal matrices with entries x_k and z_k, respectively. Note that the idealised trajectory is completely determined once a residual trajectory $t_k(\alpha)$ is chosen. Possible targets include the origin and points on the central path "closer" to the solution. Although not essential, it is common that the terminal point on the trajectory is primal-dual feasible, i.e., $r^{\text{PD}}\big(v_k(1)\big) = 0$. The choice of residual trajectory is one of the key defining features of a method and the best-known example [24] is

$$t_k(\alpha) = \begin{pmatrix} t_k^{\text{PD}}(\alpha) \\ t_k^{\text{CS}}(\alpha) \end{pmatrix} \overset{\text{def}}{=} \begin{pmatrix} (1-\alpha) r^{\text{PD}}(v_k) \\ (1-\alpha) r^{\text{CS}}(v_k) + \alpha\sigma_k \mu(v_k) e \end{pmatrix} \tag{2.9}$$

for some $\sigma_k \in (0, 1)$, which is *linear* in α.

Since r is nonlinear, an exact solution of (2.7) is generally unknown even when the residual trajectory is linear in α. Thus instead of following the solution trajectory exactly, a series approximation

$$v_{k,\ell}(\alpha) = v_k^{[0]} + \sum_{i=1}^{\ell} \frac{v_k^{[i]}}{i!}\alpha^i \equiv v_k + \sum_{i=1}^{\ell} \frac{v_k^{[i]}}{i!}\alpha^i \tag{2.10}$$

to $v_k(\alpha)$ about $\alpha = 0$ is traced, where the superscript $[i]$ denotes the i-th derivative of v_k with respect to α evaluated at $\alpha = 0$. The series coefficients are obtained by differentiating both sides of (2.7) and evaluating at $\alpha = 0$. Specifically, we have that

$$\begin{pmatrix} H & -A^T & -I \\ A & 0 & 0 \\ Z_k & 0 & X_k \end{pmatrix} \begin{pmatrix} x_k^{[1]} \\ y_k^{[1]} \\ z_k^{[1]} \end{pmatrix} = \begin{pmatrix} t_k^{\text{PD}\,[1]}(0) \\ t_k^{\text{CS}\,[1]}(0) \end{pmatrix} \tag{2.11}$$

and

$$\begin{pmatrix} H & -A^T & -I \\ A & 0 & 0 \\ Z_k & 0 & X_k \end{pmatrix} \begin{pmatrix} x_k^{[i]} \\ y_k^{[i]} \\ z_k^{[i]} \end{pmatrix} = \begin{pmatrix} t_k^{\text{PD}\,[i]}(0) \\ t_k^{\text{CS}\,[i]}(0) - \sum_{j=1}^{i-1} c_j^i x_k^{[j]} \cdot z_k^{[i-j]} \end{pmatrix} \tag{2.12}$$

for $i = 2, \ldots, \ell$, where c_i^p is the binomial coefficient "p choose i". Thus a second defining feature of a method is the type and order of series approximation taken.

The point $v_{k,\ell}(\alpha)$ is an approximation to $v_k(\alpha)$ and it is therefore unlikely that $v_{k,\ell}(1)$ will lie on the central path, or even within the required neighbourhood defined by (2.4) and (2.5). Since our aim is to reduce complementarity to zero while ultimately guaranteeing primal and dual feasibility, we pick $\alpha_{k,\ell}$ as the solution of

$$\begin{aligned} &\underset{\alpha\in(0,1]}{\text{minimize}}\ \phi\big(v_{k,\ell}(\alpha);\tau\big) \\ &\text{subject to } \gamma_{\text{L}} \le c^{\text{CS}}\big(v_{k,\ell}(\beta)\big) \le \gamma_{\text{U}} \quad \text{and} \\ &\qquad\qquad c^{\text{F}}\big(v_{k,\ell}(\beta)\big) \ge 0 \quad \text{for all} \quad 0 \le \beta \le \alpha \le 1. \end{aligned} \tag{2.13}$$

The resulting $v_{k,\ell}(\alpha_{k,\ell})$ then satisfies (2.4) and (2.5) and is measurably no worse than v_k since our aim is to minimize $\phi(v;\tau)$.

In order to guarantee convergence, we use the first-order ($\ell = 1$) Taylor approximation (2.10)–(2.11) to the idealised trajectory $v_k(\alpha)$ defined with residual trajectory (2.9) since setting $v_{k+1} = v_{k,1}^{\text{Z}} \overset{\text{def}}{=} v_{k,1}(\alpha_{k,1})$ has been shown by Zhang [51] to force $\phi(v;\tau)$ to zero when $\tau = 1$, and $\gamma_{\text{U}} = \infty$. The same is true [52] of the second-order ($\ell = 2$) Taylor approximation $v_{k,2}^{\text{Z}} \overset{\text{def}}{=} v_{k,2}(\alpha_{k,2})$.

Our algorithm views the step $v_{k,1}^{\text{Z}}$ as a sort of "Cauchy point" [5], as used by trust-region methods to ensure convergence. In particular, our general algorithm accepts

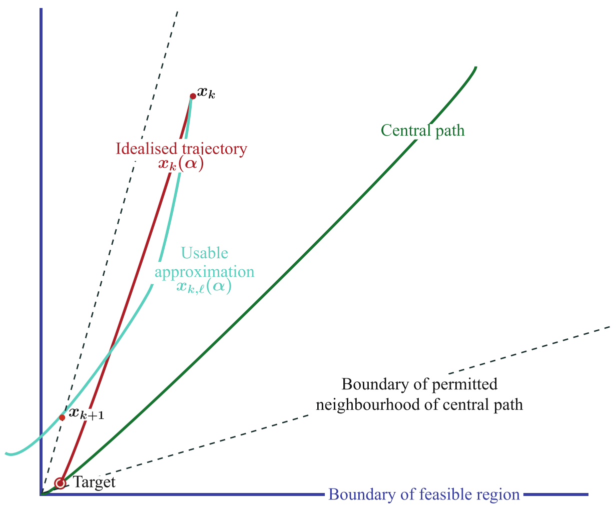

Fig. 1 Perturbed trajectory of the central path

any iterate that is "better" than $v^{\mathrm{Z}}_{k,1}$. To be more precise, v_{k+1} may be defined as any point satisfying

$$0 < (x_{k+1}, z_{k+1}), \quad \gamma_{\mathrm{L}} \le c^{\mathrm{CS}}(v_{k+1}) \le \gamma_{\mathrm{U}}, \quad 0 \le c^{\mathrm{F}}(v_{k+1}), \\ \text{and} \quad \phi(v_{k+1};\tau) \le \phi(v^{\mathrm{Z}}_{k,1};\tau). \tag{2.14}$$

Note that our strategy is well-defined since $v^{\mathrm{Z}}_{k,1}$ satisfies (2.14), and that the condition on c^{CS} implies that

$$\text{if} \quad \lim_{k\to 0} \mu(v_k) = 0 \quad \text{then} \quad \lim_{k\to\infty} x_k \cdot z_k - \mu(v_k)e = 0. \tag{2.15}$$

Obvious candidates that may accelerate convergence are $v_{k,\ell}(\alpha_{k,\ell})$ for $\ell > 1$, but we explore additional possibilities in Sect. 4. All the salient points of our algorithm are illustrated in Fig. 1.

We now formally state the algorithm.

```
Algorithm 2.1. The general algorithm.
Input $v_0 = (x_0, y_0, z_0)$ such that $(x_0, z_0) > 0$.
Choose parameters $\tau > 0$ and $0 < \sigma_{\min} \le \sigma_{\max} < 1$.
Define $0 < \gamma_{\mathrm{L}} < 1 < \gamma_{\mathrm{U}}$ and $\nu \in (0,1)$ so that $\gamma_{\mathrm{L}} \le c^{\mathrm{CS}}(v_0) \le \gamma_{\mathrm{U}}$, and $c^{\mathrm{F}}(v_0) \ge 0$.
$k \leftarrow 0$
do
    Calculate $\mu(v_k)$ and choose $\sigma_k \in [\sigma_{\min}, \sigma_{\max}]$.
    Compute $v^{\mathrm{Z}}_{k,1} = v_{k,1}(\alpha_{k,1})$ from problem (2.13).
    Compute any step $v_{k+1}$ that satisfies (2.14).
    $k \leftarrow k + 1$
end do
```

Global convergence is considered in the next section and is relatively straight forward since Algorithm 2.1 is an extension of Zhang's method [51]. However, the generality of our algorithm provides us the freedom to consider additional steps that may accelerate local *and* global convergence and is the topic of Sect. 4.

3 Convergence results

In this section we give global convergence, local convergence, and complexity results. We use results from Zhang [51] and Zhang and Zhang [52] since their algorithms are both particular instances of our method. Our first result is that Algorithm 2.1 has (at least) linear global convergence.

Theorem 1 *If* $\{v_k\}_{k=0}^{\infty}$ *is the sequence of iterates generated by Algorithm 2.1, then there exists a* $\delta \in (0, 1)$ *such that* $(x_k, z_k) > 0$ *and*

$$\phi(v_{k+1}; \tau) \le (1 - \delta)\phi(v_k; \tau)$$

for all $k \ge 0$*. Thus, if* v_* *is any limit point of the sequence of iterates, then it is a minimizer of problem QP.*

Proof The method by Zhang [51, Alg.2] is equivalent to Algorithm 2.1 for the choice $\tau = 1$, $\gamma_{\mathrm{U}} = \infty$, and $v_{k+1} = v_{k,1}^{\mathrm{Z}}$. However, one may verify that all the results in [51] and [52] still hold for any $\tau \ge 0$ and $\gamma_{\mathrm{U}} > 1$; the additional restriction $c^{\mathrm{CS}}(\alpha) \le \gamma_{\mathrm{U}}$ may change a constant contained in a bound on the minimum step length satisfying $\gamma_{\mathrm{L}} \le c^{\mathrm{CS}}(\alpha) \le \gamma_{\mathrm{U}}$ (see [51, Lem.6.3] and [52, Lem.7.1–7.3]), and the $\tau \ge 0$ is simply "carried along" in the proofs. Interestingly, this means that Zhang [51] could have used the merit function $\phi(v) = x^T z$, which corresponds to $\tau = 0$.

Zhang [51, Thm.7.1] proves that the merit function ϕ decreases linearly during each iteration. However, his algorithm makes an assumption on the initial feasible point. This mild assumption is removed by Billups and Ferris [2, Thm.2.8] since they prove that Zhang's results still hold for an arbitrary (strictly positive) starting point. From the above observations and (2.14) we conclude that

$$\begin{aligned} \phi(v_{k+1}; \tau) &\le \phi(v_{k,1}^{\mathrm{Z}}; \tau) && \text{for all} \quad k \ge 0 \\ &\le (1-\delta)\phi(v_k; \tau) && \text{and some} \quad \delta \in (0, 1). \end{aligned} \tag{3.1}$$

It also follows from (2.14) and the fact that $(x_0, z_0) > 0$ that $(x_k, z_k) > 0$ for all k, which in turn ensures that ϕ is bounded below. Thus, we deduce that $\lim_{k\to\infty} \phi(v_k; \tau) = 0$ and that Algorithm 2.1 is at least globally linearly convergent. It then follows immediately that if $v_* = \lim_{k\in\mathcal{K}} v_k$ for some subsequence $\mathcal{K} \subseteq \mathbb{N}$, then using (2.2) yields

$$\lim_{k\in\mathcal{K}} r(v_k) = r(v_*) = 0$$

so that v_* is a solution to QP. □

The following complexity result depends on the solution set

$$\mathcal{S}^* = \{v \in \mathbb{R}^{2n+m} : v \in \mathcal{F} \quad \text{and} \quad x \cdot z = 0\} \tag{3.2}$$

and the related quantity

$$\rho^* = \min\{\|v\| : v \in \mathcal{S}^*\}. \tag{3.3}$$

Theorem 2 *Let $\{v_k\}_{k=0}^{\infty}$ be the sequence of iterates generated by Algorithm 2.1 and assume that $(x_0, z_0) = \rho e$ for some number $\rho \geq \rho^*/\sqrt{n}$, that $v_{k,2}^{Z}$ is computed during each iteration, and that the next iterate is chosen as the trial point with the smallest value of ϕ. It follows that for any $t > 1$, there exists a number K_t such that*

$$\|r(v_k)\| \leq 2^{-t}\|r(v_0)\| \quad \textit{for all} \quad k \geq K_t = O(n^{3/2}t).$$

Moreover, if v_0 is strictly feasible, i.e., $v_0 \in \mathcal{F}_0$, then $K_t = O(n^{3/4}t)$.

Proof Our goal is to prove that

$$\phi(v_k; \tau) \leq 2^{-t}\phi(v_0; \tau) \quad \text{for all} \quad k \geq K_t = O(n^{3/2}t)$$

since this may be combined with the conditions in (2.14) to obtain the desired result. To this end, recall that $v_{k,1}^{Z}$ and $v_{k,2}^{Z}$ are the steps used in [51, equation (3.1)] and [52, Alg.2], respectively. Thus, it follows from the assumptions of this theorem, [51, Thm.7.1], and [52, proof of Thm.7.1] that there exist positive constants $\delta_1, \delta_2 \in (0, 1)$ such that

$$\begin{aligned}\phi(v_{k+1}; \tau) &\leq \min[\phi(v_{k,1}^{Z}; \tau), \phi(v_{k,2}^{Z}; \tau)] \\ &\leq \min[(1-\delta_1)\phi(v_k; \tau), (1-\delta_2)\phi(v_k; \tau)] \leq (1-\delta_2)\phi(v_k; \tau).\end{aligned}$$

This implies that

$$\phi(v_k; \tau) \leq (1-\delta_2)^k \phi(v_0, \tau)$$

so that the desired result holds provided $k \geq \frac{-t}{\ln(1-\delta_2)} \geq \frac{-\ln(2)t}{\ln(1-\delta_2)}$. A Taylor expansion of the logarithm yields

$$-\ln(1-\delta_2) = \delta_2 \sum_{j=0}^{\infty} \frac{1}{j+1}\delta_2^j > \delta_2.$$

Therefore, the desired result holds for all k satisfying

$$k \geq \frac{t}{\delta_2} \geq \frac{-t}{\ln(1-\delta_2)}.$$

The theorem has been proved since $\delta_2 = \Omega(1/n^{3/2})$ (see [52, Thm.7.1]). For the case $v_0 \in \mathcal{F}_0$, the bound on δ_2 becomes $\delta_2 = \Omega(1/n^{3/4})$ (see [52, Thm.8.2]). □

Algorithm 2.1 keeps $\{\sigma_k\}_{k\geq 0}$ uniformly bounded away from zero. However, it is reasonable to use the update

$$\sigma_k = \min\left(0.1, \mu(v_k)^q\right) \quad \text{for some} \quad q > 0. \tag{3.4}$$

The next theorem shows that this does not interfere with global convergence.

Theorem 3 *If $\{v_k\}_{k=0}^{\infty}$ is the sequence of iterates generated by Algorithm 2.1 with σ_k updated by (3.4), then*

$$\lim_{k\to\infty} \|r(v_k)\| = \lim_{k\to\infty} \phi(v_k;\tau) = 0.$$

Thus, if v_* is any limit point of the sequence of iterates, then it is a minimizer of QP.

Proof Suppose that there exists a number ε such that

$$\{\mu(v_k)\}_{k\in\mathcal{S}} \geq \varepsilon > 0 \quad \text{for some subsequence} \quad \mathcal{S} \subseteq \mathbb{N} \tag{3.5}$$

so that $\{\sigma_k\}_{k\in\mathcal{S}} \geq \min(0.1, \varepsilon^q)$. As in Theorem 1, it follows from Zhang [51, Thm.7.1] that there exists a $\delta > 0$ such that for all k,

$$\phi(v_k;\tau) \leq \phi(v_0;\tau) \prod_{j=0,\, j\in\mathcal{S}}^{k-1} (1-\delta) \leq \phi(v_0;\tau)(1-\delta)^{n_k} \tag{3.6}$$

where $n_k = \text{card}\{j \in \mathcal{S} : j < k\}$, card $\mathcal{B}$ denotes the cardinality of a generic set $\mathcal{B}$, and we have used the fact that $\phi(v_{k+1};\tau) \leq \phi(v_k;\tau)$ for all $k \notin \mathcal{S}$. Since $(x_k, z_k) > 0$ implies that $\phi(v_k;\tau)$ is bounded from below, we deduce from (3.6) and $\lim_{k\to\infty} n_k = \infty$ that $\lim_{k\to\infty} \phi(v_k;\tau) = 0$. Since this implies that $\lim_{k\to\infty} \mu(v_k) = 0$ we have reached a contradiction to our assumption (3.5), and conclude that $\lim_{k\to\infty} \mu(v_k) = 0$. The conditions given by (2.14) then imply that $\lim_{k\to\infty} \|r(v_k)\| = 0$ and $\lim_{k\to\infty} \phi(v_k;\tau) = 0$. The optimality of any limit point v_* follows exactly as in Theorem 1. □

4 Residual trajectories and performance enhancing steps

The main feature that determines the efficiency of our method is the choice of idealised solution trajectory $v_k(\alpha)$, which is completely determined by the choice of residual trajectory $t_k(\alpha)$ in (2.7). In this section we provide additional choices for $t_k(\alpha)$ and describe how they may be used to help accelerate convergence of our method on both nondegenerate and degenerate problems.

4.1 Lustig–Marsten–Shanno–Zhang residual trajectories

This is the case mentioned in (2.9) in which the residual trajectory $t_k(\alpha)$ is defined by

$$t_k^{\text{PD}}(\alpha) = (1-\alpha) r^{\text{PD}}(v_k) \quad \text{and} \tag{4.1a}$$

$$t_k^{\text{CS}}(\alpha) = (1-\alpha) r^{\text{CS}}(v_k) + \alpha\sigma_k\mu(v_k)e \tag{4.1b}$$

for some $\sigma_k \in (0, 1)$ [24,51]. The centering parameter σ_k is required to ensure that for small α, the trajectory turns into the neighbourhood (2.4)–(2.5) and thus progress is possible. The linearity of both $r^{\text{PD}}(v)$ and $t_k^{\text{PD}}(\alpha)$ and (2.10)–(2.12) imply that

$$\begin{aligned} r^{\text{PD}}\big(v_{k,\ell}(\alpha)\big) &= \begin{pmatrix} Hx_{k,\ell}(\alpha) + g - A^T y_{k,\ell} - z_{k,\ell} \\ Ax_{k,\ell} - b \end{pmatrix} \\ &= r^{\text{PD}}(v_k) + \alpha t_k^{\text{PD}[1]}(0) = (1-\alpha) r^{\text{PD}}(v_k) \end{aligned} \tag{4.2}$$

and, therefore, if the residual trajectory (4.1a) is used and $v_{k+1} = v_{k,\ell}(1)$, then $r^{\text{PD}}(v_j) = 0$ for all $j > k$ and $\ell \geq 1$.

A method based solely on (4.1) is likely to experience difficulties when solving degenerate problems as Sect. 1.2 clearly shows. In these situations, a reasonable alternative is a residual trajectory $t_k(\alpha)$ defined by

$$t_k^{\text{PD}}(\alpha) = (1-\alpha)^2 r^{\text{PD}}(v_k) \quad \text{and} \tag{4.3a}$$

$$t_k^{\text{CS}}(\alpha) = (1-\alpha)^2 r^{\text{CS}}(v_k) + (2\alpha - \alpha^2)\sigma_k\mu(v_k)e. \tag{4.3b}$$

This is the same trajectory as (4.1) but with a different parameterization that is given by $\alpha \mapsto 1 - (1-\alpha)^2$. This choice copes with the fact that the path v_k based on the residual trajectory (4.1) is not analytic as a solution is approached. (A Taylor series approximation to the solution $v_k(\alpha)$ of (2.7) using residual trajectory (4.3) may be interpreted as a Puiseux (square root) series [38, p.98] approximation in the parameter $1 - \sqrt{1-\alpha}$ for (2.7) using (4.1).) In contrast to (4.1), if (4.3a) is used, then the quadratic nature of $t_k^{\text{PD}}(\alpha)$ and (2.10)–(2.12) imply that

$$\begin{aligned} r^{\text{PD}}\big(v_{k,1}(\alpha)\big) &= (1-2\alpha) r^{\text{PD}}(v_k) \quad \text{and} \\ r^{\text{PD}}\big(v_{k,\ell}(\alpha)\big) &= \big(1-\alpha^2\big) r^{\text{PD}}(v_k) \quad \text{for} \quad \ell \geq 2. \end{aligned} \tag{4.4}$$

Thus if a linear approximation $v_{k,1}(\alpha)$ is used and a full step is taken, i.e., $\alpha = 1$, then there will still be no improvement in the primal-dual infeasibility. However, if the update $v_{k+1} = v_{k,\ell}(\frac{1}{2})$ or $v_{k+1} = v_{k,\ell}(1)$ for $\ell \geq 2$ is ever used, then $r^{\text{PD}}(v_j) = 0$ for all $j > k$ provided the residual trajectory (4.3a) is used.

4.2 A mixed residual trajectory

For a mixed residual trajectory we use (4.1a) to define $t_k^{\text{PD}}(\alpha)$ and (4.3b) to define $t_k^{\text{CS}}(\alpha)$. The resulting residual trajectory $t_k(\alpha)$ and idealized trajectory $v_k(\alpha)$ resolve the difficulties caused by degeneracy by recognizing that the complications arise from the complementarity slackness condition and not from the primal or dual feasibility.

4.3 Potra–Stoer–Sun–Zhao residual trajectories

The residual trajectories defined in Sects. 4.1 and 4.2 all define idealised solution trajectories satisfying $v_k(0) = v_k$ and $x_k(1) \cdot z_k(1) = \sigma_k \mu(v_k) e$, i.e., starts at v_k and leads towards a point lying "down" the central path. In this section we consider the residual trajectory $t_k(\alpha)$, where $t_k^{\text{PD}}(\alpha)$ is given by (4.1a),

$$t_k^{\text{CS}}(\alpha) = (1-\alpha)\Big(r^{\text{CS}}(v_k) + \alpha\sigma_k\mu(v_k)\big[\mu(v_k)e - r^{\text{CS}}(v_k)\big]\Big), \tag{4.5}$$

and $\sigma_k \in (0, 1]$, which defines an idealised solution trajectory that *aims for the solution*. This residual trajectory is again quadratic in α and by design initially leads into the neighbourhood defined by (2.4)–(2.5) so that progress is always possible. This trajectory with the choice $\sigma_k = 1$ is attributed to Stoer by Zhao and Sun [53], while the general σ_k case is considered by Potra and Stoer [36].

As before, a re-parameterization of the residual function can be used to ensure that the trajectory is analytic even for degenerate problems. The most obvious choice is to once again use $\alpha \mapsto 1-(1-\alpha)^2$, which is equivalent to replacing (4.1a) by (4.3a) and (4.5) by

$$t_k^{\text{CS}}(\alpha) = (1-\alpha)^2\Big(r^{\text{CS}}(v_k) + (2\alpha-\alpha^2)\sigma_k\mu(v_k)\big[\mu(v_k)e - r^{\text{CS}}(v_k)\big]\Big) \tag{4.6}$$

[36], which is quartic in α. However, the simpler

$$t_k^{\text{CS}}(\alpha) = (1-\alpha)^2\Big(r^{\text{CS}}(v_k) + \alpha\sigma_k\mu(v_k)\big[\mu(v_k)e - r^{\text{CS}}(v_k)\big]\Big) \tag{4.7}$$

is also possible [53] and is only cubic in α.

We close this section with the following observation: to ensure primal and dual feasibility are obtained when $\alpha = 1$, a method based on (4.5), (4.6), or (4.7) must use at least a 2nd, 4th, or 3rd degree Taylor approximation, respectively.

5 Implementation

In this section we describe the main features of our new GALAHAD [13] Fortran 2003 convex quadratic programming package CQP. Further details, including a complete description of all user control parameters, are provided in the package documentation provided as part of GALAHAD.[1]

5.1 Preprocessing

CQP accepts problems of the form (1.1). In practice, variables and constraints are reordered internally by the package so that those with (just) lower bounds occur before

[1] Available from http://galahad.rl.ac.uk/galahad-www/. A Matlab interface is also provided.

those that are bounded from both sides which themselves occur before those that are (just) bounded from above. This leads to more effective vector operations when (for example) checking for feasibility since there is then no need to constantly re-check what sort of bounds a given constraint has. Equality constraints may be mixed with the inequalities, and the preprocessing phase implicitly separates them from the inequalities. Fixed variables and dependent equality constraints are also removed. In addition, slack variables c are introduced for general inequality constraints. Thus at an internal level, CQP actually solves problems of the form

$$\textbf{CQP}_\textbf{I}: \quad \underset{x,c}{\text{minimize}} \; q(x) \quad \text{subject to} \quad A^{\text{E}}x = c^{\text{E}}, \; A^{\text{I}}x - c = 0, \; x^{\text{L}} \le x \le x^{\text{U}} \quad \text{and} \quad c^{\text{L}} \le c \le c^{\text{U}},$$

where only the first parts of x^{L} and c^{L} and the last parts of x^{U} and c^{U} are nonzero. The KKT conditions for CQP_I are that

$$\begin{array}{l} Hx_* + g = (A^{\text{E}})^T y_*^{\text{E}} + (A^{\text{I}})^T y_*^{\text{I}} + z_*^{\text{L}} - z_*^{\text{U}} \quad \text{and} \quad y_*^{\text{I}} = y_*^{\text{L}} - y_*^{\text{U}} \\ A^{\text{E}}x_* = c^{\text{E}} \quad \text{and} \quad A^{\text{I}}x_* - c_* = 0, \quad \text{and} \\ (c_* - c^{\text{L}}) \cdot y_*^{\text{L}} = 0, \quad (c^{\text{U}} - c_*) \cdot y_*^{\text{U}} = 0, \quad (x_* - x^{\text{L}}) \cdot z_*^{\text{L}} = 0 \quad \text{and} \quad (x^{\text{U}} - x_*) \cdot z_*^{\text{U}} = 0 \end{array} \tag{5.1}$$

for primal variables $x^{\text{L}} \le x_* \le x^{\text{U}}$, constraint values $c^{\text{L}} \le c_* \le c^{\text{U}}$, Lagrange multipliers y_*^{E}, y_*^{I} and $(y_*^{\text{L}}, y_*^{\text{U}}) \ge 0$, and dual variables $(z_*^{\text{L}}, z_*^{\text{U}}) \ge 0$. In what follows, we write $v = (x, c, y^{\text{E}}, y^{\text{I}}, y^{\text{L}}, y^{\text{U}}, z^{\text{L}}, z^{\text{U}})$, and use the obvious notation v_*, v_k and $v_{k,\ell}(\alpha)$ for the solution vector, the vector of iterates and the approximate solution trajectory, etc. Some components of v, such as those components of z_*^{L} for which x is not bounded below, are identically zero; the package automatically recognises this and the relevant components are not stored.

In view of (5.1), we note that now

$$r^{\text{PD}}(v) = \begin{pmatrix} Hx + g - A^{\text{E}T} y^{\text{E}} - A^{\text{I}T} y^{\text{I}} - z^{\text{L}} + z^{\text{U}} \\ y^{\text{I}} - y^{\text{L}} + y^{\text{U}} \\ A^{\text{E}}x - c^{\text{E}} \\ A^{\text{I}}x - c \end{pmatrix}, \quad r^{\text{CS}}(v) = \begin{pmatrix} (c - c^{\text{L}}) \cdot y^{\text{L}} \\ (c^{\text{U}} - c) \cdot y^{\text{U}} \\ (x - x^{\text{L}}) \cdot z^{\text{L}} \\ (x^{\text{U}} - x) \cdot z^{\text{U}} \end{pmatrix}$$

and

$$\mu(v) = \frac{(c - c^{\text{L}})^T y^{\text{L}} + (c^{\text{U}} - c)^T y^{\text{U}} + (x - x^{\text{L}})^T z^{\text{L}} + (x^{\text{U}} - x)^T z^{\text{U}}}{n_\text{I}},$$

where the total number of inequality constraints $x^{\text{L}} \le x \le x^{\text{U}}$ and $c^{\text{L}} \le c \le c^{\text{U}}$ is n_I.

5.2 Dependent constraints

To detect dependent equality constraints, we use the GALAHAD module FDC. This offers two alternatives especially geared towards large problems. The first [3] finds a sparse symmetric indefinite factorization

$$\begin{pmatrix} \alpha I & (A^{\mathrm{E}})^T \\ A^{\mathrm{E}} & 0 \end{pmatrix} = LBL^T$$

using GALAHAD's module SLS (see Sect. 5.5) and assesses rank deficiency from tiny eigenvalues of the one-by-one and two-by-two diagonal blocks that make up B—a value $\alpha = 0.01$ is used by default. The second, default, alternative is simply to perform a sparse unsymmetric rectangular factorization

$$A^{\mathrm{E}} = LU \quad \text{or (by default)} \; (A^{\mathrm{E}})^T = LU$$

using threshold pivoting, and to identify tiny "diagonal" entries from the "lower triangular" matrix L obtained [8]. We recognise that neither is as robust as, for example, a singular-value decomposition or a rank-revealing QR factorization, but both have proved reliable in our tests.

5.3 Getting started

Despite numerous attempts to provide a clever "starting" point v_0, our current preferred strategy is simply to set v_0 a "large distance" from any of its lower and upper bounds l and u. Simply, by default we set each component a distance $\min(\omega, \frac{1}{2}(u-l))$ from its nearest bound with $\omega = 10^4$ being our favourite choice. Our experience is that it is often far easier to come in "from arbitrary infinity", than to move to the central path from an initial point that tries too cleverly to predict which (and start near) bounds might be active.

5.4 Residual trajectories

CQP offers the linear and quadratic residual trajectories (4.1) and (4.3), the mixed residual trajectory mentioned in Sect. 4.2, the Potra–Stoer–Sun–Zhao residual trajectories (4.1a)/(4.5) and (4.3a)/(4.7) as well as a variant that switches from the former to the latter once $\mu(v_k)$ is smaller than 10^{-4}.

5.5 Linear system solver

Having selected a residual trajectory, the next task is to compute the usable approximation $v_{k,\ell}(\alpha)$ in (2.10) to the idealised trajectory. We offer the choice of using a single approximation for a specified ℓ, or that of using every $v_{k,i}(\alpha)$ for $i \leq \ell$, since the coefficients for each $v_{k,i}$ are trivially a subset of those for $v_{k,\ell}$. If we are not using the linear residual trajectory (4.1), we will additionally have to compute $v^{\mathrm{Z}}_{k,1}$ as our guarantee of convergence.

At each iteration, we need to solve the analogs of the linear systems (2.11)–(2.12) for CQP$_{\mathrm{I}}$ to determine the coefficients of the expansions $v_{k,\ell}(\alpha)$ we will use. These are

$$\begin{pmatrix} H & 0 & -A^{\mathrm{E}T} & -A^{\mathrm{I}T} & 0 & 0 & -I & I \\ 0 & 0 & 0 & I & -I & I & 0 & 0 \\ A^{\mathrm{E}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A^{\mathrm{I}} & -I & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & Y_k^{\mathrm{L}} & 0 & 0 & C_k - C^{\mathrm{L}} & 0 & 0 & 0 \\ 0 & -Y_k^{\mathrm{U}} & 0 & 0 & 0 & C^{\mathrm{U}} - C_k & 0 & 0 \\ Z_k^{\mathrm{L}} & 0 & 0 & 0 & 0 & 0 & X_k - X^{\mathrm{L}} & 0 \\ -Z_k^{\mathrm{U}} & 0 & 0 & 0 & 0 & 0 & 0 & X^{\mathrm{U}} - X_k \end{pmatrix} \begin{pmatrix} x_k^{[i]} \\ c_k^{[i]} \\ y_k^{\mathrm{E}\,[i]} \\ y_k^{\mathrm{I}\,[i]} \\ y_k^{\mathrm{L}\,[i]} \\ y_k^{\mathrm{U}\,[i]} \\ z_k^{\mathrm{L}\,[i]} \\ z_k^{\mathrm{U}\,[i]} \end{pmatrix} = \begin{pmatrix} r_k^{\mathrm{X}\,[i]} \\ r_k^{\mathrm{Y}\,[i]} \\ r_k^{\mathrm{E}\,[i]} \\ r_k^{\mathrm{I}\,[i]} \\ r_k^{\mathrm{CL}\,[i]} \\ r_k^{\mathrm{CU}\,[i]} \\ r_k^{\mathrm{XL}\,[i]} \\ r_k^{\mathrm{XU}\,[i]} \end{pmatrix} \tag{5.2}$$

for generic right-hand sides $r^{[i]}$ and where once again C_k, C^{L} (etc.) are diagonal matrices with entries c_k and c^{L}. Rather than solving (5.2) directly, we use the relationships

$$\begin{aligned} &(C_k - C^{\mathrm{L}}) y_k^{\mathrm{L}\,[i]} = r_k^{\mathrm{CL}\,[i]} - Y_k^{\mathrm{L}} c_k^{[i]}, \quad (C^{\mathrm{U}} - C_k) y_k^{\mathrm{U}\,[i]} = r_k^{\mathrm{CU}\,[i]} + Y_k^{\mathrm{U}} c_k^{[i]}, \\ &(X_k - X^{\mathrm{L}}) z_k^{\mathrm{L}\,[i]} = r_k^{\mathrm{XL}\,[i]} - Z_k^{\mathrm{L}} x_k^{[i]} \text{ and } (X^{\mathrm{U}} - X_k) z_k^{\mathrm{U}\,[i]} = r_k^{\mathrm{XU}\,[i]} + Z_k^{\mathrm{U}} x_k^{[i]} \end{aligned} \tag{5.3}$$

to eliminate $y_k^{\mathrm{L}\,[i]}$, $y_k^{\mathrm{U}\,[i]}$, $z_k^{\mathrm{L}\,[i]}$ and $z_k^{\mathrm{U}\,[i]}$, which results in the symmetric, indefinite system

$$\begin{pmatrix} \begin{matrix} H + (X_k - X^{\mathrm{L}})^{-1} Z_k^{\mathrm{L}} \\ +(X^{\mathrm{U}} - X_k)^{-1} Z_k^{\mathrm{U}} \end{matrix} & 0 & A^{\mathrm{E}T} & A^{\mathrm{I}T} \\ 0 & \begin{matrix} (C_k - C^{\mathrm{L}})^{-1} Y_k^{\mathrm{L}} \\ +(C^{\mathrm{U}} - C_k)^{-1} Y_k^{\mathrm{U}} \end{matrix} & 0 & -I \\ A^{\mathrm{E}} & 0 & 0 & 0 \\ A^{\mathrm{I}} & -I & 0 & 0 \end{pmatrix} \begin{pmatrix} x_k^{[i]} \\ c_k^{[i]} \\ -y_k^{\mathrm{E}\,[i]} \\ -y_k^{\mathrm{I}\,[i]} \end{pmatrix} = \begin{pmatrix} \begin{matrix} r_k^{\mathrm{X}\,[i]} + (X_k - X^{\mathrm{L}})^{-1} r_k^{\mathrm{XL}\,[i]} \\ -(X^{\mathrm{U}} - X_k)^{-1} r_k^{\mathrm{XU}\,[i]} \end{matrix} \\ \begin{matrix} r_k^{\mathrm{Y}\,[i]} + (C_k - C^{\mathrm{L}})^{-1} r_k^{\mathrm{CL}\,[i]} \\ -(C^{\mathrm{U}} - C_k)^{-1} r_k^{\mathrm{CU}\,[i]} \end{matrix} \\ r_k^{\mathrm{E}\,[i]} \\ r_k^{\mathrm{I}\,[i]} \end{pmatrix}. \tag{5.4}$$

Once we have found $x_k^{[i]}$, $c_k^{[i]}$, $y_k^{\mathrm{E}\,[i]}$ and $y_k^{\mathrm{I}\,[i]}$ from (5.4), we trivially recover $y_k^{\mathrm{L}\,[i]}$, $y_k^{\mathrm{U}\,[i]}$ $z_k^{\mathrm{L}\,[i]}$ and $z_k^{\mathrm{U}\,[i]}$ from the diagonal systems (5.3).

CQP uses another GALAHAD module, SBLS, to solve symmetric block systems

$$\begin{pmatrix} G & B^T \\ B & -E \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}, \tag{5.5}$$

for which G and E are symmetric, like (5.4). In fact, SBLS simply gathers the data G, B, E, a and b to set up one of a number of possible relevant systems for solution by yet another GALAHAD module, SLS. In particular, SBLS provides either a *full-space* approach, in which (5.5) is simply solved by forming and factorizing

$$K = \begin{pmatrix} G & B^T \\ B & -E \end{pmatrix},$$

a *range-space approach* in which factorizations of G and the Schur complement $E + BG^{-1}B^T$ are used, or a *null-space* approach in which a non-singular sub-matrix of B is determined and used to implicitly construct a basis matrix N for the null-space of B and ultimately rests on needing to factorize the positive-definite matrix $N^T GN$. The range-space approach is currently limited to the case where G is a non-singular diagonal (but in principle could be adapted for block diagonal G), while the null-space approach is restricted for simplicity to the case where $E = 0$ [as is the case for (5.4)]. Such ideas are well known [10, Section 5.4] and their applicability generally depends on the relative ratio of the row and column dimension of B; the range-space method is best when B has relatively few rows, the null space one when the ratio is close to (but no greater than) one, and the full-space approach when neither of these occurs. The default in CQP is to use the range-space approach whenever G is non-singular and diagonal, and the full-space approach otherwise.

The module SLS called by SBLS provides a common interface to a number of freely-available and commercial software sparse, symmetric direct solvers—currently these include `MA27`, `MA57`, `MA77`, `MA86`, `MA87` and `MA97` from HSL [19], `PARDISO` from the Pardiso Project [37] and `WSMP` from the IBM alpha Works [17]. Options available include a variety of sparse ordering and scalings, iterative refinement, partial, out-of-core and parallel solution, and inertia determination and correction. Further efficiencies are possible when the matrix is definite (such as would occur for the null- and range-space approaches of concern here).

5.6 Staying feasible

Armed with one or more search arcs $v_{k,\ell}(\alpha)$ from (2.10), it remains to find the stepsize $\alpha_{k,\ell}$ that gives the solution to (2.13). We do this by first determining $\alpha^{\rm L}_{k,\ell}$ as the largest $\alpha \in (0, 1]$ for which

$$\mu\big(v_{k,\ell}(\alpha)\big) - \nu\|r^{\rm PD}\big(v_{k,\ell}(\alpha)\big)\| \geq 0 \quad \text{and} \tag{5.6a}$$
$$\gamma_{\rm L}\mu\big(v_{k,\ell}(\alpha)\big)e \leq r^{\rm CS}\big(v_{k,\ell}(\alpha)\big) \leq \gamma_{\rm U}\mu\big(v_{k,\ell}(\alpha)\big)e \quad \text{for all} \quad \alpha \in [0, \alpha^{\rm L}_{k,\ell}] \tag{5.6b}$$

and then using a simple backtracking Armijo line-search[2] to find $\alpha_{k,\ell}$ as the first in the sequence $\{2^{-i}\alpha^{\rm L}_{k,\ell}\}_{i\geq 0}$ for which

$$\phi\big(v_{k,\ell}(\alpha);\tau\big) \leq \phi\big(v_k;\tau\big) + \eta\alpha\phi^{[1]}\big(v_k;\tau\big)$$

for a given small $\eta \in (0, 1)$ and as before the bracketed superscript denotes the first derivative; default values $\nu = 10^{-5}$, $\gamma_{\rm L} = 10^{-5}$, $\gamma_{\rm U} = \infty$, $\tau = 1$ and $\eta = 10^{-4}$ are used. Since our aim is actually simply to satisfy (2.14), we first calculate $v^{\rm Z}_{k,1}$, and then

[2] When $\ell \leq 2$, the exact line minimizer of $\phi\big(v_{k,\ell}(\alpha);\tau\big)$ is found by calculus.

for each usable approximation $v_{k,\ell}(\alpha)$ we have chosen, we compute $\alpha_{k,\ell}$ and compare $\phi(v(\alpha_{k,\ell}); \tau)$ with $\phi(v^{\rm Z}_{k,1}; \tau)$. The smallest overall defines v_{k+1}.

It is straightforward to show that

$$\mu^{[1]}(v_k) = e^T t^{\rm CS\,[1]}(0)/n_{\rm I} \tag{5.7}$$

—this is by analogy with the special case of problem CQP for which (2.11) gives (componentwise) $z_k \cdot x^{[1]}_k + x_k \cdot z^{[1]}_k = t^{\rm CS\,[1]}(0)$ from which (5.7) follows since for CQP $\mu^{[1]}(v_k) = e^T(z_k \cdot x^{[1]}_k + x_k \cdot z^{[1]}_k)/n$—and that

$$\left.\frac{d\|r^{\rm PD}(v_{k,\ell}(\alpha))\|}{d\alpha}\right|_{\alpha=0} = \frac{r^{{\rm PD}\,T}(v_k) t^{\rm PD\,[1]}(0)}{\|r^{\rm PD}(v_k)\|} \tag{5.8}$$

when $r^{\rm PD}(v_k) \neq 0$—the derivative for general α is simply

$$\begin{aligned}\frac{d\|r^{\rm PD}(v_{k,\ell}(\alpha))\|}{d\alpha} &= \frac{r^{\rm PD}\ T(v_{k,\ell}(\alpha)) r^{\rm PD\,[1]}(v_{k,\ell}(\alpha))}{\|r^{\rm PD}(v_{k,\ell}(\alpha))\|} \\ &= \frac{r^{\rm PD}\ T(v_{k,\ell}(\alpha)) \nabla r^{\rm PD}(v_{k,\ell}(\alpha)) v^{[1]}_{k,\ell}(\alpha)}{\|r^{\rm PD}(v_{k,\ell}(\alpha))\|}\end{aligned}$$

which gives the required result since

$$\nabla r^{\rm PD}(v_{k,\ell}(0)) v_{k,\ell}{}^{[1]}(0) = \nabla r^{\rm PD}(v_k) v_{k,\ell}{}^{[1]}(0) = \nabla r^{\rm PD}(v_k) v^{[1]}_k = t^{\rm PD\,[1]}(0)$$

because of (2.11). Thus

$$\phi^{[1]}\big(v_k; \tau\big) = e^T t^{\rm CS\,[1]}(0) + \tau\delta(r^{\rm PD}(v_k)) \frac{r^{\rm PD}\ T(v_k) t^{\rm PD\,[1]}(0)}{\|r^{\rm PD}(v_k)\|} \tag{5.9}$$

where $\delta(r) = 1$ if $r \neq 0$ and 0 if $r = 0$.

Note that for the linear residual trajectory (4.1),

$$\phi^{[1]}\big(v_k; \tau\big) = -\phi\big(v_k; \tau\big) + n_{\rm I}\sigma_k \mu(v_k) \quad \text{and} \quad \|r^{\rm PD}\big(v_{k,\ell}(\alpha)\big)\| = (1-\alpha)\|r^{\rm PD}\big(v_k\big)\|$$

because of (4.2) and (5.9), while for the quadratic residual trajectory (4.3),

$$\begin{gathered}\phi^{[1]}\big(v_k; \tau\big) = -2\phi\big(v_k; \tau\big) + \ 2n_{\rm I}\sigma_k \mu(v_k) \qquad \text{and} \\ \|r^{\rm PD}\big(v_{k,\ell}(\alpha)\big)\| = \begin{cases} (1-2\alpha)\|r^{\rm PD}\big(v_k\big)\| & \text{for} \quad \ell = 1 \\ (1-\alpha)^2\|r^{\rm PD}\big(v_k\big)\| & \text{for} \quad \ell > 1 \end{cases}\end{gathered}$$

from (4.4) and (5.9).

The required inequality (5.6a) and the $2n_{\rm I}$ inequalities (5.6b) each require that we find the smallest root of a scalar polynomial of degree 2ℓ in (0, 1] (if any). For $\ell = 1$

and 2, it is easiest simply to enumerate the roots from stable versions of the well-known formulae for quadratic and quartic equations. For larger ℓ, rather than enumerate roots, we instead use Sturm's theorem (via the Euclidean algorithm) to count the number of roots in an iteratively-bisected interval until we find a bounding interval containing the single desired root. Thereafter, a safeguarded Newton iteration is used to refine this root [9]. This has two essential advantages. Firstly, if (as is often the case) there is no root in the initial interval, this is discovered very fast. Secondly, since as we wish to find the smallest positive root of all the inequalities (5.6), for successive inequalities, we only need investigate $[0, \alpha_c]$, where α_c is the current champion. Early detection of a small α_c leads to small subsequent intervals and thus efficiencies as in our first point. We take precautions since the Euclidean algorithm is somewhat prone to break down; if we become aware of large generated coefficients, we simply resort to a more expensive but robust method that computes all the real roots [25]. All of this functionality is available as part of GALAHAD's ROOTS module.

An additional expense is the computation of the required polynomial coefficients for $\mu\big(v_{k,\ell}(\alpha)\big)$ and $r^{\text{CS}}\big(v_{k,\ell}(\alpha)\big)$. Generically each of these is obtained from the products of two polynomials of order ℓ.

5.7 Parallelism

While many aspects of our algorithm (matrix-vector products, distances to bounds, residuals) might be calculated in parallel, our experience with shared-memory systems like OpenMP is simply that the start-up costs usually outweigh any benefits from parallel execution—an optimizing compiler often appears to perform just as good a job as hand-crafted parallelism. However, we have found it beneficial to offer parallel directives for the matrix factorization/system solves and the polynomial root extraction. The former is provided naturally by the component solvers `MA77`, `MA86`, `MA87`, `MA97`, `PARDISO` and `WSMP` and is generally very useful. Indeed our experience is that parallel factorization, at least on small systems (say 4–16 processors as might occur on a modern PC), is so successful that other less demanding sequential tasks become significant. In particular, we find that computing linear system solutions (forward and backward substitution) now becomes an issue, simply because this phase is limited by memory bandwidth [18]. The polynomial solution described in Sect. 5.6 is essentially independent and thus simply and effectively embedded within a parallel do loop. There is a small loss in efficiency since the best α_c mentioned earlier is now per-processor rather than global. We had hoped that the calculation of the polynomial coefficients $r^{\text{CS}}\big(v_{k,\ell}(\alpha)\big)$ would also benefit from parallelism, but in our experience start-up costs again dominated.

5.8 Presolve and problem scaling

CQP is one of a number of quadratic programming solvers available with GALAHAD. Since each has essentially the same interface, a common wrapper module `QP` is also provided that adds further functionality. In particular `QP` provides a presolve/preprocessing phase that aims to simplify a problem prior to solution using the GALAHAD module PRESOLVE [16], and a variety of pre-scalings that try to equilibrate H, A and the problem bounds to try to make the solution phase easier. Since these options were not enabled in our tests, we provide no further details here.

5.9 Termination

We stop the algorithm as soon as

$$\|r^{\mathrm{I}}(v_k)\| \leq \max(\varepsilon^{\mathrm{A}}, \varepsilon^{\mathrm{R}}\|r^{\mathrm{I}}(v_0)\|) \quad \text{for each of I} \in \{\text{P, D, CS}\}$$

for given small $\varepsilon^{\mathrm{A}}, \varepsilon^{\mathrm{R}} > 0$, or when a maximum iteration limit is reached. We also check for infeasibility by monitoring $\phi(v_k; \tau)$ and stop if stagnation over a given number of iterations seems to have occcured.

6 Numerical experiments

Our tests involve the quadratic programming examples from the combined CUTEr [12] and Maros and Meszaros [26] test sets—multiple instances of very similar nature have been excluded. We considered 178 convex examples in all (see Table A.1 in on-line Appendix A), of which 51 are reported as degenerate; see on-line Appendix B for the settings used to decode the problems. All experiments were performed on four cores of a Dell Precision T340, single Core2 Quad Q9550 processor(2.83GHz, 1333MHz FSB, 12MB L2 Cache) with 4GB RAM; CQP and its dependencies are in double precision and compiled with gfortran 4.3 using fast (-O3) optimization and OpenMP enabled (-fopenmp).

We have compared variants of the codes using two different residual trajectories and for each two different degrees of series approximation to the solution trajectory. We considered the linear and quadratic residual trajectories (4.1) and (4.3). For the former we tried the first and second degree Taylor approximations recommended by Zhang [51] and Zhang and Zhang [52]; we refer to these as variants T1 and T2. For the latter, we used approximations of degrees two and four (P2 and P4); as we suggested in Sect. 4.1, a first-degree approximation would not improve the primal-dual infeasibility for the unit steplength.

For our tests, we use default values for most control parameters. The exceptions are that we choose the linear solver MA97 for SLS with two iterative refinements per solve, the absolute and relative stopping tolerances are set to $\varepsilon^{\mathrm{A}} = \varepsilon^{\mathrm{R}} = 10^{-5}$, at most 1,000 iterations are allowed and at most 20 iterations are permitted for which ϕ fails to decrease by 0.1 before the problem is declared primal or dual infeasible; see on-line Appendix C for details.

The complete results from these experiments are summarised in Tables A.2–A.5 in the on-line appendix. To make comparisons easier, we provide a graphical interpretation of this data using the performance profiles of the factorization counts and run time in Figs. 2 and 3; briefly, given a set of test problems and a set of competing algorithms, the i-th performance profile $p_i(\omega)$ indicates the fraction of problems for which the i-th algorithm is within a factor ω of the best for a given metric—see [6]

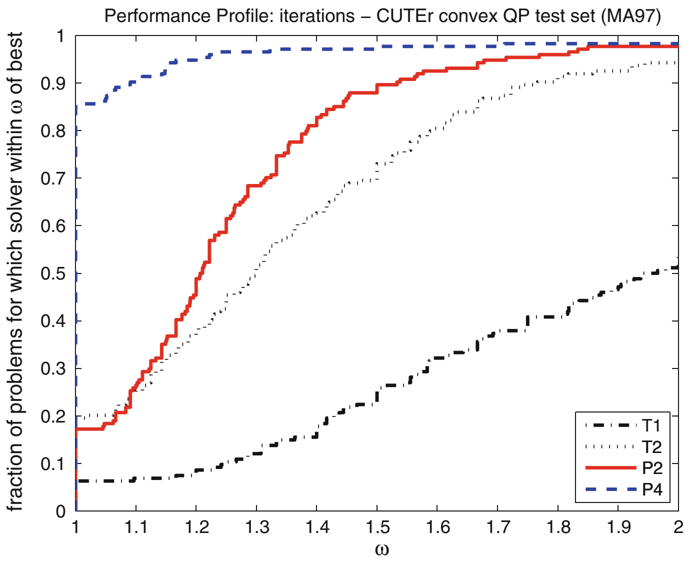

Fig. 2 Performance profile for the number of iterations (factorizations) required to solve the sample set of CUTEr problems using T1, T2, P2 and P4. Variants T1, T2, P2 and P4 failed to achieve the required accuracy for 5, 4, 3 and 3 of the 178 test problems, respectively

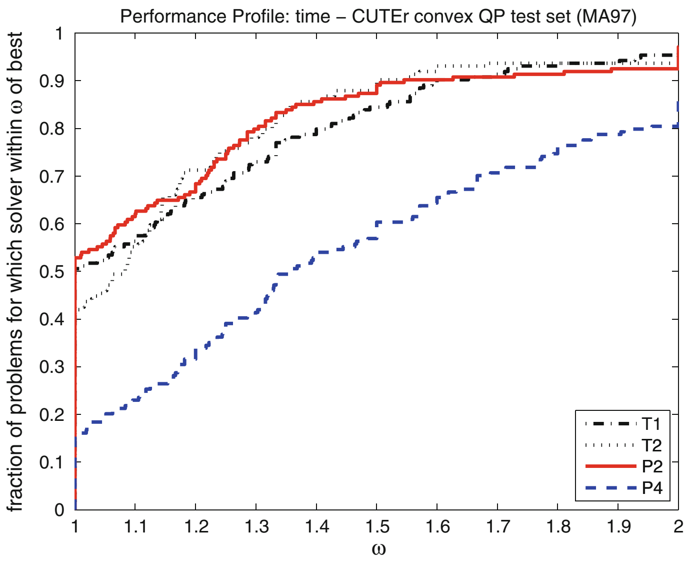

Fig. 3 Performance profile for the time spent in solving the sample set of CUTEr problems using T1, T2, P2 and P4

for a formal definition of performance profiles and a discussion of their properties. In Figs. 4 and 5 we focus simply on the subset of 51 degenerate problems.

In Fig. 2, we clearly see the advantage of using higher-order polynomial approximations to the solution trajectories. More accurate approximations lead to fewer

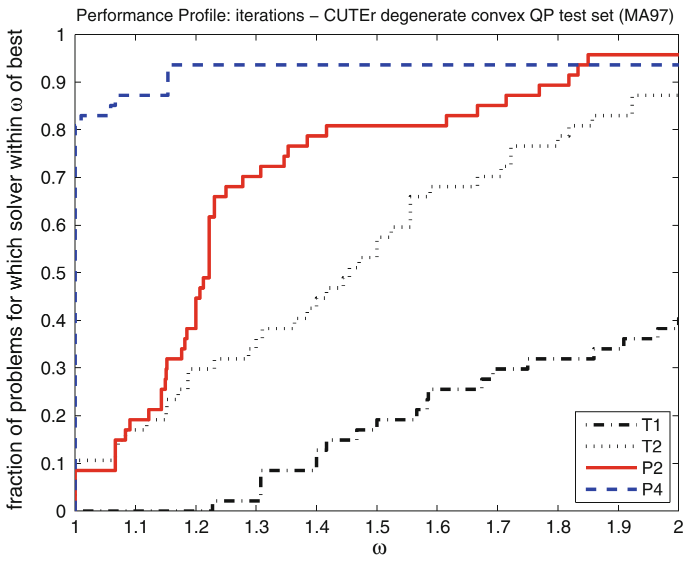

Fig. 4 Performance profile for the number of iterations (factorizations) required to solve the degenerate subset of 51 CUTEr problems using T1, T2, P2 and P4. Variants T1, T2, P2 and P4 failed to achieve the required accuracy for 4, 4, 2 and 3 of the 51 degenerate test problems, respectively

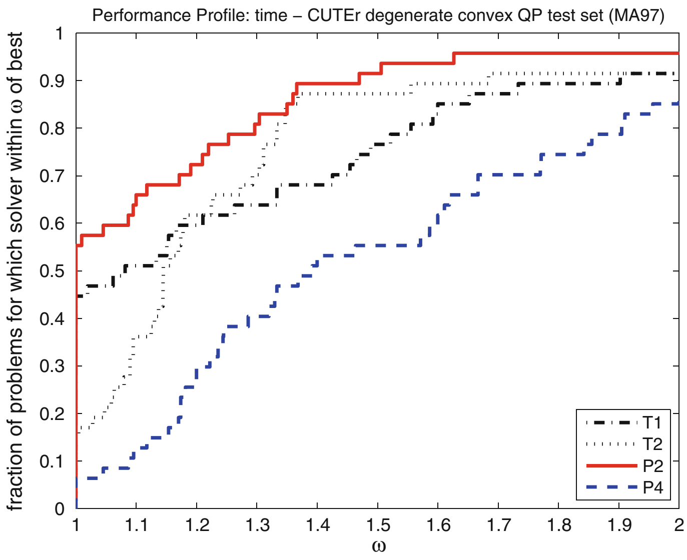

Fig. 5 Performance profile for the time spent in solving the degenerate subset of 51 CUTEr problems using T1, T2, P2 and P4

iterations/factorizations. The curves T2 and P2 indicate a slight but not overwhelming advantage of using the quadratic residual trajectory in comparison to the linear one. This trend is more pronounced in Fig. 4 in which we focus on the degenerate examples. We note that in a few cases the required stopping accuracy was not attained.

Table 1 High accuracy solution

Name	T1			T2			P2			P4		
	Iter	Time	Error	Iter	Time	Error	Iter	Time	Error	Iter	Time	Error
DEGDIAG	28	2.37	4.4^{-5}	20	2.80	3.8^{-5}	15	2.33	9.7^{-8}	12	3.54	3.8^{-17}
DEGTRID	26	3.44	2.7^{-4}	18	4.15	3.9^{-4}	12	3.69	1.3^{-12}	8	3.48	9.7^{-11}
DEGTRID2	34	4.52	6.0^{-7}	24	5.32	6.8^{-7}	5	1.27	7.6^{-8}	5	2.15	6.3^{-7}
DEGTRIDL	193	29.03	6.0^{-6}	80	18.84	1.3^{-6}	11	2.94	3.3^{-9}	7	3.18	1.5^{-10}

When it comes to run times, however, Fig. 3 shows that this reduction in the number of factorizations is counterbalanced by the need to solve additional systems and to find roots of polynomial equations. In particular, recall that every method needs to find at least $v^{Z}_{k,1}$, and for T1 this is all that is required. The cost in P4 clearly more than offsets the gains in iterations. When we focus on the degenerate examples in Fig. 5, the advantage of using the quadratic residual trajectory is apparent, and here it is reasonable to suggest that P2 performs best.

One of our stated aims was to show that it is possible to solve even degenerate examples accurately. The solutions to most of our test problems are not known exactly, so we illustrate how our methods work on a few (contrived) large examples, `DEGDIAG`, `DEGTRID`, `DEGTRID2` and `DEGTRIDL`, from the test set, for which the exact solution is available. We again consider the T1, T2, P2 and P4 variants, but now set the absolute and relative stopping tolerances to $\varepsilon^{\mathrm{A}} = \varepsilon^{\mathrm{R}} = 10^{-12}$; again, see on-line Appendix C for details. In Table 1 we give the number of iterations required, the time taken and the maximum error in x for each variant. As we see, the quadratic residual trajectory pays off both in terms of effort and of attained accuracy for these examples.

We have experimented with the other residual trajectories mentioned in Sect. 5.4 with mixed success. Although each proves to perform well (and sometimes exceptionally so) in some cases, none proves to be as reliable as the simple linear and quadratic ones we have focused on here.

7 Conclusions and discussion

Relatively little is known about the existence of the central path when strict complementarity does not hold. For convex optimization problems and a decreasing sequence of parameters $\{\mu_k\}$ converging to zero, minimizers of the log-barrier function, i.e., solutions to (1.5), are known to converge to a constrained minimizer [47]. For proper convex objective functions and linear constraints, the existence and uniqueness of the central path is established in [7] without requiring strict complementarity but assuming that the set of constrained minimizers is nonempty and bounded. This path has the desirable property that for any $\bar{\mu} > 0$, the path parametrized over $(0, \bar{\mu}]$ is compact and can be extended continuously to a constrained minimizer of the original problem as $\mu \to 0$. When strict complementarity fails, the central path seems to be best understood for convex quadratic programming (more precisely for monotone and sufficient linear complementarity problems [42,43]). The key to the analysis is to re-parametrize the path using $\sqrt{\mu}$ in place of μ. Whether similar results hold for more general problem classes such as nonconvex quadratic optimization or nonlinear optimization, is still an open question; some clues appear in [27] and [49], but again numerical experience is scarce. However, the non-quadratic extension of Example 2 in Sect. 1.2 given by

$$\underset{x}{\text{minimize}} \ \tfrac{1}{p}x^p \quad \text{subject to} \quad x \geq 0$$

for fixed $p \geq 1$, for which $x(\mu) = \mu^{\frac{1}{p}}$ and $z(\mu) = \mu^{\frac{p-1}{p}}$, leaves no doubt as to the need for a non-Taylor trajectory in some cases—of course there is no guarantee that a general nonlinear problem has (local or global) central path(s) without strong additional assumptions such as strict complementarity or second-order sufficiency.

Example 2 in Sect. 1.2 is unfortunately typical in that it produces an approximate solution with only a few digits of accuracy. Obtaining accurate solutions to degenerate convex quadratic programs is important in its own right, but is also crucial in certain active-set-based sequential quadratic programming methods for nonlinear optimization. For such methods, Table 1 suggests that the P2 and P4 strategies are particularly attractive.

Variants on the path-following methods proposed here are possible. For instance, rather than using Taylor Series approximations of idealised trajectories, one might instead fit a Puiseux expansion of the form

$$a_0 + a_{\frac{1}{2}}\mu^{\frac{1}{2}} + a_1\mu + a_{\frac{3}{2}}\mu^{\frac{3}{2}} + a_2\mu^2 + \cdots$$

to a prescribed number of past iterates. Convergence of such a procedure is unclear, but it may also be effective in handling degenerate problems.

In this paper we have not considered the rate of local convergence. However, it is reasonable to assume that the iterates will converge superlinearly provided (i) sufficiently high Taylor approximations to the idealized trajectories are utilised and (ii) a residual trajectory defined by (4.1a) and (4.5) is used, or if (4.1a) and (4.1b) are used and $\sigma_k > 0$ is decreased to zero sufficiently fast. Indeed, in practice we routinely observe superlinear convergence.

We now *informally* argue how results from Potra and Stoer [36] may be used to prove fast local convergence for a slightly modified version of Algorithm 2.1. Assume that Algorithm 2.1 is executed such that for all iterates suficiently large the target trajectory defined by (4.1a) and (4.5) is used so that

$$t_k^{\text{CS}}(\alpha) = (1-\alpha)\mu(v_k)\left(\frac{r^{\text{CS}}(v_k)}{\mu(v_k)} + \alpha\sigma_k\big[\,\mu(v_k)e - r^{\text{CS}}(v_k)\,\big]\right). \tag{7.1}$$

We also must slightly modify our algorithm so that during each iteration the "wide" neighborhood (2.4) is widened further, as described by [36, Equations (3.3), (3.4), (3.11), (3.40)]. Then, to apply the results of [36, Thm.2.2] and deduce that all derivatives given by (2.11) and (2.12) are bounded, we must show that the vector

$$\frac{r^{\text{CS}}(v_k)}{\mu(v_k)} + \alpha\sigma_k\big[\,\mu(v_k)e - r^{\text{CS}}(v_k)\,\big] = \left(\frac{1}{\mu(v_k)} - \alpha\sigma_k\right) x_k \cdot z_k + \alpha\sigma_k\mu(v_k)e$$

used in (7.1) is contained within a compact subset of $\{x \in \mathbb{R}^n : x > 0\}$ for all $\alpha \in (0, 1]$ and all k sufficiently large. This can easily be shown using (3.4), the definition of $\mu(v_k)$, the fact that $\lim_{k\to\infty} \mu(v_k) = 0$, (2.3), and (2.4). Since this is the key result used by Potra and Stoer to obtain their results, we are not surprised that Algorithm 2.1 consistently exhibits superlinear convergence.

Acknowledgments This work was supported by the EPSRC grants EP/E053351/1, EP/F005369/1 and EP/H026053/1 and NSERC Discovery Grant 299010-04. We are extremely grateful to a referee and associate editor for comments that lead to important clarifications of our numerical results.

References

1. Anitescu, M., Lesaja, G., Potra, F.A.: Equivaence between different formulations of the linear complementarity problem. Optim. Methods Softw. **7**(3–4), 265–290 (1997)
2. Billups, S.C., Ferris, M.C.: Convergence of an infeasible interior-point algorithm from arbitrary positive starting points. SIAM J. Optim. **6**(2), 316–325 (1996)
3. Cartis, C., Gould, N.I.M.: Finding a point in the relative interior of a polyhedron. Technical Report RAL-TR-2006-016, Rutherford Appleton Laboratory. Chilton (2006)
4. Colombo, M., Gondzio, J.: Further development of multiple centrality correctors for interior point methods. Comput. Optim. Appl. **41**(3), 277–305 (2008)
5. Conn, A.R., Gould, N.I.M., Toint, Ph.L.: Trust-Region Methods. MPS-SIAM Series on Optimization. SIAM publications, Philadelphia (2000)
6. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**(2), 201–213 (2002)
7. Graña Drummond, L.M., Svaiter, B.F.: On well definedness of the central path. J. Optim. Theory Appl. **102**(2), 223–237 (1999)
8. Duff, I.S., Reid, J.K.: The design of MA48: a code for the direct solution of sparse unsymmetric linear systems of equations. Trans. ACM Math. Softw. **22**(2), 187–226 (1996)
9. Dunaway, D.K.: Calculation of zeros of a real polynomial through factorization using Euclid's algorithm. SIAM J. Numer. Anal. **11**(6), 1087–1104 (1974)
10. Gill, P.E., Murray, W., Wright, M.H.: Practical Optimization. Academic Press, London (1981)
11. Gondzio, J.: Multiple centrality corrections in a primal-dual method for linear programming. Comput. Optim. Appl. **6**(2), 137–156 (1996)
12. Gould, N.I.M., Orban, D., Toint, Ph.L.: CUTEr (and SifDec), a constrained and unconstrained testing environment, revisited. Trans. ACM Math. Softw. **29**(4), 373–394 (2003)
13. Gould, N.I.M., Orban, D., Toint, Ph.L.: GALAHAD—a library of thread-safe fortran 90 packages for large-scale nonlinear optimization. Trans. ACM Math. Softw. **29**(4), 353–372 (2003)
14. Gould, N.I.M., Robinson, D.P.: A second derivative SQP method: global convergence. SIAM J. Optim. **20**(4), 2023–2048 (2010)
15. Gould, N.I.M., Robinson, D.P.: A second derivative SQP method: local convergence and practical issues. SIAM J. Optim. **20**(4), 2049–2079 (2010)
16. Gould, N.I.M., Toint, Ph.L.: Preprocessing for quadratic programming. Math. Program. B **100**(1), 95–132 (2004)
17. Gupta, A.: WSMP: Watson sparse matrix package part I—direct solution of symmetric sparse system. Research Report RC 21886. IBM T. J. Watson Research Center, Yorktown Heights (2010)
18. Hogg, J.D., Scott, J.A.: A note on the solve phase of a multicore solver. Technical Report RAL-TR-2010-007, Rutherford Appleton Laboratory, Chilton (2010)

19. HSL. A collection of Fortran codes for large-scale scientific computation (2011). http://www.hsl.rl.ac.uk
20. Ji, J., Potra, F.A., Huang, S.: Predictor–corrector method for linear complementarity problems with polynomial complexity and superlinear convergence. J. Optim. Theory Appl. **85**(1), 187–199 (1995)
21. Kojima, M., Mizuno, S., Noma, T.: Limiting behavior of trajectories generated by a continuation method for monotone complementarity problems. Math. Oper. Res. **15**(4), 662–675 (1990)
22. Liu, X., Potra, F.A.: Predictor-corrector methods for sufficient linear complementarity problems in a wide neighborhood of the central path. Optim. Methods Softw. **20**(1), 145–168 (2005)
23. Liu, X., Potra, F.A.: Corrector–predictor methods for sufficient linear complementarity problems in a wide neighborhood of the central path. SIAM J. Optim. **17**(3), 871–890 (2006)
24. Lustig, I.J., Marsten, R.E., Shanno, D.F.: Computational experience with a primal–dual interior point method for linear programming. Linear Algebra Appl. **152**, 191–222 (1991)
25. Madsen, K., Reid, J.K.: Fortran subroutines for finding polynomial zeros. Technical Report AERE-R 7986, AERE Harwell Laboratory, Harwell (1975)
26. Maros, I., Meszaros, C.: A repository of convex quadratic programming problems. Optim. Methods Softw. **11–12**, 671–681 (1999)
27. McCormick, G.P., Witzgall, C.: Logarithmic SUMT limits in convex programming. Math. Program. **90**(1), 113–145 (2001)
28. Megiddo, N.: Pathways to the optimal set in linear programming. In: Megiddo, N. (ed.) Progress in Mathematical Programming, pp. 131–158. Springer, New-York (1989)
29. Mehrotra, S.: On the implementation of a primal–dual interior point method. SIAM J. Optim. **2**, 575–601 (1992)
30. Mizuno, S.: A superlinearly convergent infeasible-interior-point algorithm for geometrical LCPs without a strictly complementary condition. Math. Oper. Res. **21**(2), 382–400 (1996)
31. Mizuno, S., Todd, M.J., Ye, Y.: On adaptive-step primal–dual interior-point algorithms for linear programming. Math. Oper. Res. **18**, 964–981 (1993)
32. Monteiro, R.D.C., Tsuchiya, T.: Limiting behavior of the derivatives of certain trajectories associated with a monotone horizontal linear complementarity problem. Math. Oper. Res. **21**(4), 793–814 (1996)
33. Potra, F.A.: A superlinearly convergent predictor–corrector method for degenerate LCP in a wide neighborhood of the central path with $O(\sqrt{n}L)$-iteration complexity. Math. Program. **100**(2), 317–337 (2004)
34. Potra, F.A.: Primal–dual affine scaling interior point methods for linear complementarity problems. SIAM J. Optim. **19**(1), 114–143 (2008)
35. Potra, F.A., Sheng, R.: Superlinearly convergent infeasible-interior-point algorithm for degenerate lcp. J. Optim. Theory Appl. **97**(2), 249–269 (1998)
36. Potra, F.A., Stoer, J.: On a class of superlinearly convergent polynomial time interior point methods for sufficient LCP. SIAM J. Optim. **20**(3), 1333–1363 (2009)
37. Schenk, O., Gärtner, K.: On fast factorization pivoting methods for symmetric indefinite systems. Electron. Trans. Numer. Anal. **23**, 158–179 (2006)
38. Siegel, C.L.: Topics in Complex Function Theory. Elliptic Functions and Uniformization Theory, vol. 1. Wiley, Chichester (1988)
39. Stoer, J.: High order long-step methods for solving linear complemenarity problems. Ann. Oper. Res. **103** (2001)
40. Stoer, J.: Analysis of interior-point paths. J. Res. Natl. Inst. Stand. Technol. **111**(2) (2006)
41. Stoer, J., Wechs, M.: The complexity of high-order predictor-corrector methods for solving sufficient linear complementarity problems. Optim. Methods Softw. **10**(2), 393–417 (1998)
42. Stoer, J., Wechs, M.: Infeasible-interior-point paths for sufficient linear complementarity problems and their analyticity. Math. Program. **83**(1–3), 407–423 (1998)
43. Stoer, J., Wechs, M.: On the analyticity properties of infeasible-interior-point paths for monotone linear complementarity problems. Numer. Math. **81**, 631–645 (1999)
44. Stoer, J., Wechs, M., Mizuno, S.: High order infeasible-interior-point methods for solving sufficient linear complementarity problems. Math. Oper. Res. **23**(4), 832–862 (1998)
45. Sturm, J.F.: Superlinear convergence of an algorithm for monotone linear complementarity problems, when no strictly complementary solution exists. Math. Oper. Res. **24**(1), 72–94 (1999)
46. Väliaho, H.: P_*-matrices are just sufficient. Linear Algebra Appl. **239**, 103–108 (1996)
47. Wright, M.H.: Interior methods for constrained optimization. Acta Numer. **1**, 341–407 (1992)

48. Wright, S.J.: Primal–Dual Interior-Point Methods. Society for Industrial and Applied Mathematics (SIAM), Philadelphia (1997)
49. Wright, S.J., Orban, D.: Local convergence of the Newton/log-barrier method for degenerate problems. Math. Oper. Res. **27**(3), 585–613 (2002)
50. Ye, Y., Güler, O., Tapia, R.A., Zhang, Y.: A quadratically convergent $O(\sqrt{n}L)$-iteration algorithm for linear programming. Math. Program. **59**, 151–162 (1993)
51. Zhang, Y.: On the convergence of a class of infeasible interior-point methods for the horizontal linear complementarity problem. SIAM J. Optim. **4**(1), 208–227 (1994)
52. Zhang, Y., Zhang, D.: On polynomiality of the Mehrotra-type predictor–corrector interior-point algorithms. Math. Program. **68**, 303–318 (1995)
53. Zhao, G., Sun, J.: On the rate of local convergence of high-order-infeasible-path- following algorithms for P_*-linear complementarity problems. Comput. Optim. Appl. **14**(3), 293–307 (1999)