Comput Optim Appl
DOI 10.1007/s10589-007-9160-7

A bundle-type algorithm for routing in telecommunication data networks

Claude Lemaréchal · Adam Ouorou · Georgios Petrou

Received: 31 October 2006 / Revised: 11 December 2007

Abstract To optimize the quality of service through a telecommunication network, we propose an algorithm based on Lagrangian relaxation. The bundle-type dual algorithm is adapted to the present situation, where the dual function is the sum of a polyhedral function (coming from shortest path problems) and of a smooth function (coming from the congestion function).

Keywords Convex optimization · Routing · Multicommodity flows · Kleinrock delay function

1 Introduction

1.1 The model

We consider the following problem

$$\min\ f(y) := \sum_{j=1}^{n} f_j(y_j)$$

C. Lemaréchal (✉)
INRIA Rhônes-Alpes, 655 avenue de l'Europe, Montbonnot, 38334 Saint Ismier, France
e-mail: claude.lemarechal@inria.fr

A. Ouorou
France Telecom R&D, CORE/MCN, 38-40 rue du Général Leclerc, 92794 Issy-Les-Moulineaux cedex 9, France

G. Petrou
EDF R&D, Osiris, 1 av. du Général de Gaulle, 92141 Clamart, France

$$\begin{aligned} \text{s.t.} \quad & Ax^k = b^k \in \mathbb{R}^m, \quad k = 1, \ldots, K, \\ & \sum_{k=1}^{K} x^k = y \in \mathbb{R}^n, \\ & 0 \le y < c, \quad x^k \ge 0, \ k = 1, \ldots, K, \end{aligned} \tag{1.1}$$

where

- A is the node-arc incidence matrix of a graph $G = (V, E)$ (m nodes, n arcs),
- x^k are flow vectors representing K commodities between source nodes s_k and sink nodes t_k, $k = 1, \ldots, K$,
- $b^k \in \mathbb{R}^m$ are vectors with two nonzero components (corresponding to an origin s_k and destination t_k) such that $\sum_{i=1}^m b_i^k = 0$,
- y is the total link flow vector,
- f_j is proportional to the Kleinrock average delay function:

$$f_j(y_j) = \frac{y_j}{c_j - y_j} \quad \text{if } 0 \le y_j < c_j \text{ (and } +\infty \text{ otherwise)}, \tag{1.2}$$

- $c \in \mathbb{R}^n$ is the vector of arc capacities.

Problem (1.1), often called *multicommodity flow*, occurs in data communication networks and plays an important role in the optimization of network performances. We mention here that the delay function may assume other forms than (1.1). Our approach is significant only when the f_j's are *nonlinear*. Another remark which will be technically useful is that the feasible flows form a bounded set: they cannot exceed the capacities.

1.2 Numerical solution methods: outline

Various methods have been proposed in the literature to solve the multicommodity flow problem. Let us cite for example [25, 27], we refer to [22, 24] for a more complete review. These methods can be classified according to three basic paradigms:

(i) Direct methods exploit the problem's block structure. Most popular is flow deviation [8] because of its simplicity; it is a special case of the Frank-Wolfe method [6], which works on a sequence of linearized problems. It has slow convergence and many authors have tried to improve it.

(ii) Other classical mathematical programming algorithms (Newton, conjugate gradient) have been adapted to the structure of (1.1); see [2] for example.

(iii) Some proposals adopt a dual point of view: Lagrangian relaxation is applied to the constraints linking x and y. This results in a concave dual function to be maximized, which can be done by a suitable algorithm such as proximal, ACCPM, subgradient, ... This approach was originally proposed in [7], see also [16] for a recent study.

Judged from a nonlinear optimization point of view, methods of type (i) suffer serious convergence deficiencies. We illustrate them on Fig. 1, which assumes $K = 1$

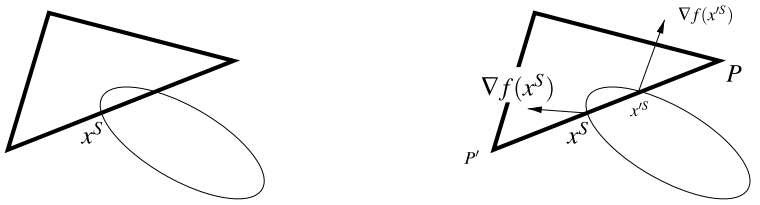

Fig. 1 Instability of Frank-Wolfe

for simplicity (then $y = x$). The left part of the picture displays the polyhedron $\{Ax = b, x \geq 0\}$, as well as the level set $f(x) = f(x^S)$ passing through the current iterate x^S, supposedly close to optimality. The essential idea of flow deviation is to linearize f at x^S, so that (1.1) becomes a linear program (let us neglect the difficulty coming from the constraint $x < c$). Now the right part of Fig. 1 shows that the solution of this LP may jump from one extreme point to another, called P and P' on the picture, if x^S moves to x'^S; and P' may be far from P, even if x^S and x'^S are close together. The effect of this *instability* on the convergence is disastrous, as is demonstrated in [28]: if $\bar{f}$ is the optimal value, we typically have $f(x^S) - \bar{f} \simeq 1/S$.

Approximating each f_j to first order is thus not accurate enough, and this motivates methods of type (ii), based on second-order approximations.

As for methods of type (iii), their motivation is the decomposable structure of (1.1). The resulting solution algorithm is made of two parts. One (minimizing the Lagrangian) treats each flow separately; the other (maximizing the Lagrangian dual) has a complexity depending only on the number n of arcs in the network. Now the aim of the present work is to introduce in this second part the second-order approximation that flow deviation is lacking. A similar idea was given quite recently by [1]; maximizing the dual by the ACCPM algorithm was then tremendously improved, both in terms of power (solving problems with n and m up to 10^5 and K up to 10^6) and of speed (CPU times divided by hundreds).

1.3 The proposed algorithm

The crucial ingredient for the methods of class (iii) is the algorithm maximizing the dual function. Here we do for bundle what [1] does for ACCPM, in a manner which can be explained as follows.

A standard approach to maximize a concave function—call it $\theta(u)$—is *cutting planes* [3, 12], in which θ is iteratively approximated by richer and richer polyhedral functions $\hat{\theta}$. These $\hat{\theta}$ are successively maximized; but this results in instabilities. A possible stabilization mechanism (the proximal bundle idea) appends to $\hat{\theta}$ a quadratic term centered at some "favored" iterate $\hat{u}$ (essentially the best current iterate).

Here, θ contains a well-isolated smooth component: $\theta = \Phi + \Pi$, where Φ is (concave and) twice differentiable, while Π is indeed polyhedral. We therefore use cutting planes to approximate Π *only*; stabilization is obtained thanks to the quadratic approximation of Φ at $\hat{u}$: a definitely natural quadratic term.

This approach can also be explained with no reference to the proximal paradigm. Maximizing a function $\theta(u)$ requires a *model* of θ, valid around the current iterate $\hat{u}$. Here, each component of θ has a natural model:

- the second-order quadratic approximation is best suited for the smooth function Φ, as in Newton's method;
- the cutting-plane approximation is natural for the polyhedral function Π, as in Kelley's method.

Our approach thus appears as quite natural, as it totally eliminates the need for a (somewhat artificial) proximal stabilization. By contrast, [1] keeps intact the (just as artificial) interior-point stabilization.

The paper is organized as follows. We propose in Sect. 2 a (classical) Lagrangian relaxation of (1.1) and in Sect. 3 our model of the dual function θ; it uses a second-order oracle for Φ and an ordinary first-order oracle for Π. Our adaptation of the bundling technique to cope with this special model is the subject of Sect. 4, while Sect. 5 recalls the aggregation mechanism, important for primal recovery. The algorithm is detailed in Sect. 6, its convergence is established in Sect. 7, while Sect. 8 shows how to recover the primal optimal solution from the dual algorithm. Finally, Sect. 9 gives some numerical results and we conclude in Sect. 10 with a general discussion of our approach.

2 Lagrangian relaxation

As already mentioned, the material developed in this section goes back to [7]. Associating with the coupling constraints $\sum_k x^k = y$ the dual variables $u \in \mathbb{R}^n$, we define the Lagrangian

$$L(x,y,u) = \sum_{j=1}^{n} f_j(y_j) + \sum_{j=1}^{n} u_j \left(-y_j + \sum_{k=1}^{K} x_j^k \right)$$

and we apply Lagrangian relaxation, as explained for example in [17]. We minimize $L(\cdot,\cdot,u)$ for fixed u; here, this amounts to computing

$$\Phi_j(u_j) := \min_{0 \le y_j < c_j} \{ f_j(y_j) - u_j y_j \}, \quad j = 1, \ldots, n, \tag{2.1}$$

$$\Pi^k(u) := \min\{ u^\top x^k : Ax^k = b^k, x^k \ge 0 \}, \quad k = 1, \ldots, K \tag{2.2}$$

(note that $\Phi_j = -f_j^*$, where f^* is the convex conjugate of the function f). It will be convenient for the sequel to use the notation

$$\Phi(u) := \sum_{j=1}^{n} \Phi_j(u_j), \qquad \Pi(u) := \sum_{k=1}^{K} \Pi^k(u).$$

The dual problem is then to maximize with respect to u the so-called *dual function*, namely: solve

$$\max_{u \in \mathbb{R}^n} \theta(u), \quad \text{where } \theta(u) := \Phi(u) + \Pi(u). \tag{2.3}$$

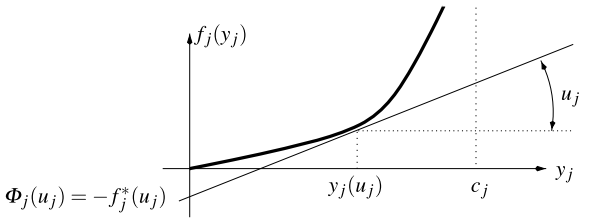

Fig. 2 Conjugating Kleinrock's function

In (2.1), the optimal y_j is easy to compute (see Fig. 2): we obtain

$$y_j(u_j) = \begin{cases} c_j - \sqrt{\frac{c_j}{u_j}} & \text{if } u_j \geq \frac{1}{c_j}, \\ 0 & \text{otherwise,} \end{cases} \tag{2.4}$$

so that Φ_j has the expression

$$\Phi_j(u_j) = \begin{cases} -(\sqrt{c_j u_j} - 1)^2 & \text{if } u_j \geq \frac{1}{c_j}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.5}$$

On the other hand, computing Π from (2.2) amounts to solving K independent shortest path problems, each of which being posed between s_k and t_k and having arc lengths u_j. The next simple result says that this computation has to be done with *positive* arc lengths only.

Proposition 2.1 *Consider the set*

$$U := \left\{ u \in \mathbb{R}^n : u_j \geq \frac{1}{c_j},\ j = 1, \ldots, n \right\}. \tag{2.6}$$

For any $u \notin U$, there is $u' \in U$ such that $\theta(u') \geq \theta(u)$. As a result, (2.3) is not changed if the constraint $u \in U$ is inserted.

Proof This is Proposition 2 in [7]. □

Thus, to ease the computation of the Π^k's, the constraints $u_j \geq 1/c_j$ may be inserted into (2.3): this does not prevent the computation of a dual optimal solution and does not change the optimal dual value.

3 Model of the dual function

Taking advantage of Proposition 2.1, we reformulate (2.3) as

$$\max_{u \in U} \theta(u) := \Phi(u) + \Pi(u) := \sum_{j=1}^{n} \Phi_j(u_j) + \sum_{k=1}^{K} \Pi^k(u). \tag{3.1}$$

To solve it, we propose a hybrid method working as follows:

– Each smooth function Φ_j is approximated by its second-order development, as in Newton's method. This development is made at a point—call it $\hat{u}$—controlled according to its (dual) objective value $\theta(\hat{u})$.
– Each polyhedral function Π^k is approximated by cutting planes, as in Kelley's method [3, 12].

In a way, the above method can be viewed as a bundle variant [18] (see also [1]), in which

– the cutting-plane paradigm is applied to a part of the (dual) objective function, namely Π,
– stabilization around $\hat{u}$ is obtained by the Newtonian term $u^\top \nabla^2 \Phi(\hat{u}) u$, instead of an artificial $\|u\|^2$ weighted by a hard-to-tune penalty coefficient.

Since Lagrangian relaxation is column generation, our algorithm can also be viewed as a Dantzig-Wolfe variant where the masters are suitably stabilized.

At each iteration s, (2.2) is solved with the iterate u^s; this provides a shortest path $x^k(u^s)$, which in turn provides an upper linearization: by definition, $\Pi^k(u) \le u^\top x^k(u^s)$ for all $u \in \mathbb{R}^n$. Accumulating these shortest paths, we form at the current iteration S the K polyhedral functions

$$\hat{\Pi}^k(u) := \min_{s=1,\dots,S} u^\top x^k(u^s) \ge \Pi^k(u), \quad \text{for all } u \in \mathbb{R}^n. \tag{3.2}$$

As for the Φ_j's, note that they have analytic derivatives over the feasible domain:

$$\Phi_j'(u_j) = \sqrt{\frac{c_j}{u_j}} - c_j, \quad \Phi_j''(u_j) = \frac{-1}{2u_j}\sqrt{\frac{c_j}{u_j}} \quad \text{if } u_j > \frac{1}{c_j} \tag{3.3}$$

and that $-\Phi_j'(u_j) = y_j(u_j)$ is the optimal y_j of (2.4).

In addition to the $\hat{\Pi}^k$'s, suppose also that a *stability center* $\hat{u} \in U$ is available at the current iteration. Analogously to Π^k, each Φ_j will be replaced by its quadratic approximation near $\hat{u}$:

$$\tilde{\Phi}_j(u_j) := \Phi_j(\hat{u}_j) - y_j(\hat{u}_j)(u_j - \hat{u}_j) + \frac{1}{2} M_j (u_j - \hat{u}_j)^2 \ [\simeq \Phi_j(u_j)], \tag{3.4}$$

where $y_j(\hat{u}_j) = -\Phi_j'(\hat{u}_j)$ is given by (2.4) and $M_j := \Phi_j''(\hat{u}_j)$ by (3.3).

We will use the notation

$$\tilde{\Phi}(u) := \sum_{j=1}^{n} \tilde{\Phi}_j(u_j), \qquad \hat{\Pi}(u) := \sum_{k=1}^{K} \hat{\Pi}^k(u), \qquad \hat{\theta}(u) := \tilde{\Phi}(u) + \hat{\Pi}(u)$$

and the essential part of an iteration will be to maximize $\hat{\theta}$, which can be viewed as a *model* of the true dual function θ in (3.1). We also find it convenient to use the change of variable $h = u - \hat{u}$: we solve

$$\max\{\tilde{\Phi}(\hat{u} + h) + \hat{\Pi}(\hat{u} + h) : \hat{u} + h \ge 1/c\}.$$

Introducing additional variables π^k (connoting $\hat{\Pi}^k$), this is the quadratic programming problem

$$\begin{aligned} &\max_{h,\pi}\left\{\tilde{\Phi}(\hat{u}+h)+\sum_{k=1}^{K}\pi^k\right\}\\ &\text{s.t.}\quad \pi^k \le (\hat{u}+h)^\top x^k(u^s), \quad \text{for } \begin{cases} k=1,\dots,K,\\ s=1,\dots,S,\end{cases}\\ &\qquad h_j \ge \frac{1}{c_j}-\hat{u}_j, \quad j=1,\dots,n. \end{aligned} \tag{3.5}$$

Note that the somewhat artificial constraint $\hat{u}+h=u\ge 1/c$ is useful for a fast computation of $\Pi^k(u)$ in (2.2); but it is even more useful for the dual algorithm: from (2.5), $\Phi_j''(u_j)=0$ if $u_j<1/c_j$. If such a u is a $\hat{u}$, then $\tilde{\Phi}_j$ will degenerate and (3.5) will perhaps be unbounded from above.[1]

Proposition 3.1 *Problem* (3.5) *has a unique optimal solution* $(\hat{h},\hat{\pi})$, *with* $\hat{\pi}^k=\hat{\Pi}^k(\hat{u}+\hat{h})$.

Proof From standard convex analysis, each function $\hat{\Pi}^k$ is concave; and each $\tilde{\Phi}_j$ is a strictly concave quadratic function (from (3.3), $M_j<0$!): $\hat{\theta}$ has a unique maximum $\hat{u}+\hat{h}$, making up the h-part of the optimal solution in (3.5); and each π^k has to reach its maximal value, namely $\hat{\Pi}^k(\hat{u}+\hat{h})$. □

Note to conclude this section that our approach can of course be applied to the maximization of any concave function θ made up of two parts: a polyhedral one (given by an oracle) and a smooth one (whose Hessian is at hand). In (3.4), M_j is the jjth entry of the Hessian (here diagonal) of the smooth part of θ.

4 Ascent steps, null-steps and backtracking

The resolution of (3.5) *predicts* an increase

$$\delta := \hat{\theta}(\hat{u}+\hat{h})-\theta(\hat{u}) \tag{4.1}$$

in the dual objective function. Of course, $\delta\ge 0$ since $\theta(\hat{u})\le\hat{\theta}(\hat{u})\le\hat{\theta}(\hat{u}+\hat{h})$. Besides, we will see in Proposition 5.3 that δ is an optimality measure of $\hat{u}$: it is natural

[1] This difficulty can be eliminated, though. Observe in (1.1) that the constraint $y\ge 0$ is redundant; each f_j of (1.2) can therefore be extended as we like on $\mathbb{R}_-$. A convenient extension is

$$f_j(y_j) := \frac{y_j^2}{c_j^2}+\frac{y_j}{c_j} \quad \text{for } y_j\le 0,$$

which is C^2 and strongly convex; its conjugate enjoys the same properties and the algorithm can work.

to stop the algorithm if δ is small. So δ is indeed positive unless the algorithm is about to stop.

Standard bundle compares the actual increase $\theta(\hat{u}+\hat{h})-\theta(\hat{u})$ to δ; if it is deemed insufficient, (3.5) is solved again with an enriched model; this is the *bundling* process. For reasons that will soon become apparent, this technique needs amendment. In our variant, the candidate for the next iteration is not the output $\hat{u}+\hat{h}$ from (3.5) but rather $u^+ := \hat{u}+t\hat{h}$, for some suitable stepsize $t \in]0,1]$ and a sufficient increase is quantified by

$$[\theta(u^+) =]\ \theta(\hat{u}+t\hat{h}) \geq \theta(\hat{u}) + \kappa t\delta, \tag{4.2}$$

$\kappa \in]0,1[$ being a fixed tolerance. If (4.2) holds, $\hat{u}$ can safely be moved to the definitely better point u^+; iteration S is terminated, this is an *ascent step* in the bundle terminology. Theorem 7.4 will show that infinitely many such updates do imply convergence of the algorithm.

Now assume (4.2) does not hold. Because $\hat{\theta}$ is concave,

$$\begin{aligned}\hat{\theta}(u^+) &\geq (1-t)\hat{\theta}(\hat{u}) + t\hat{\theta}(\hat{u}+\hat{h})\\ &\geq (1-t)\theta(\hat{u}) + t\hat{\theta}(\hat{u}+\hat{h})\\ &= \theta(\hat{u}) + t\delta.\end{aligned}$$

Failure of (4.2) therefore implies $\hat{\theta}(u^+) > \theta(u^+) + (1-\kappa)t\delta$; this means that $\hat{\theta}$ approximates θ badly. Then we have to decide which of the approximations $\tilde{\Phi}$ and $\hat{\Pi}$ needs improvement.

Improvement of $\hat{\Pi}$ is done by the bundling process, which appends in the definition of $\hat{\Pi}$ the new data coming from the oracle (2.2), called at $\hat{u}+t\hat{h}$. However, bundling will be of no avail if $(\hat{h},\hat{\pi})$ is still feasible in the next QP (3.5)—see Lemma 7.5 below. To avoid an infinite loop, a sufficient improvement must be required on the next $\hat{\Pi}$; this is the whole business of a bundle method. We quantify the required improvement as

$$\Pi(\hat{u}+t\hat{h}) \leq \Pi(\hat{u}) + t[\hat{\Pi}(\hat{u}+\hat{h}) - \Pi(\hat{u})] - \kappa' t\delta, \tag{4.3}$$

$\kappa' \in]0,\kappa]$ being another positive tolerance (on Fig. 3, (4.3) holds when $\Pi(u^+)$ lies under the point marked A). By concavity of $\hat{\Pi}$, this implies that $\Pi(u^+)$ is "substantially" smaller than $\hat{\Pi}(u^+)$; and note that the next $\hat{\Pi}$ is going to have the value

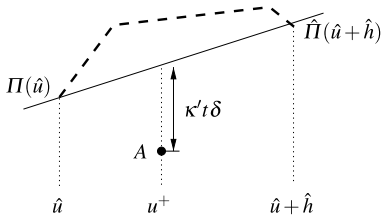

Fig. 3 The new linearization is "substantially" smaller than $\hat{\Pi}$ at u^+

$\Pi(u^+)$ at u^+. If (4.3) holds, $\hat{u}$ is kept as it is; again iteration S is terminated, this is a *null-step*.

When (4.3) does not hold, $\hat{\Pi}(u^+)$ can be considered as a good approximation of $\Pi(u^+)$; so if (4.2) does not hold, it is $\tilde{\Phi}$ that approximates Φ badly. To improve it, we decrease t, compute the new value of θ and test (4.2), (4.3) again; we will make sure in Lemma 7.1 below that this *backtracking phase* cannot go forever.

Let us summarize this section. An iteration of our variant solves (3.5) and tests $u^+ = \hat{u} + t\hat{h}$, with three possible outputs:

- if (4.2) holds, an ascent step is made: $\hat{u}$ is moved to u^+ and the model $\hat{\Pi}$ is updated;
- if (4.2) does not hold but (4.3) holds, a null-step is made: $\hat{u}$ is kept as it is and $\hat{\Pi}$ is updated;
- if neither (4.2) nor (4.3) holds, a backtracking is made: t is decreased and the oracle (2.2) is called at the new u^+, which is closer to $\hat{u}$.

Standard bundle has no backtracking: it merely uses $t = 1$ and overlooks (4.3). Actually, if we had $\theta = \Pi$ and $\hat{\theta} = \hat{\Pi}$, (4.3) could not hold when (4.2) does not hold. In the present variant, this argument is destroyed by the $\tilde{\Phi}$-part of $\hat{\theta}$.

5 Toward primal recovery: the aggregate linearization

The necessary material to state the dual algorithm is now available. However, remember that our problem is rather (1.1) than (2.3). To establish the connection between the two resolutions, we need some more sophisticated material from convex analysis. First we introduce some notation.

- The normal cone $\mathrm{N}_U(u)$ is the set of vectors $v \in \mathbb{R}^n$ such that $(v-u)^\top v \le 0$ for all $v \in U$;
- $y(\hat{u}) \in \mathbb{R}^n$ will be the vector whose components are $y_j(\hat{u}_j)$, see (2.4);
- $M := \nabla^2\Phi(\hat{u})$ will be the diagonal (negative definite) $n \times n$ matrix whose jjth element is $M_j = \Phi_j''(\hat{u}_j)$ of (3.4);
- the unit simplex of $\mathbb{R}^S$ will be $\Delta^S := \{\alpha \in \mathbb{R}^S : \sum_s \alpha^s = 1, \alpha \ge 0\}$.

Now we recall some elementary subdifferential calculus. Denote by

$$6\theta(u) := -\partial(-\theta)(u) = \{g \in \mathbb{R}^n : \theta(v) \le \theta(u) + (v-u)^\top g \text{ for all } v \in \mathbb{R}^n\}$$

the "superdifferential" of the concave function θ at u.

- The superdifferential of the smooth concave function $\tilde{\Phi}$ is its gradient: $6\tilde{\Phi}(u) = -y(\hat{u}) + M(u-\hat{u})$;
- the superdifferential of $\hat{\Pi}^k$ is the convex hull of the active slopes in (3.2):

$$6\hat{\Pi}^k(u) = \left\{\hat{x}^k = \sum_{s=1}^{S} \alpha^s x^k(u^s) : \alpha \in \Delta^S, \alpha^s = 0 \text{ if } u^\top x^k(u^s) > \hat{\Pi}^k(u)\right\}; \quad (5.1)$$

- the superdifferential of $\hat{\theta}$ is the sum of superdifferentials:

$$6\hat{\theta}(u) = -y(\hat{u}) + M(u - \hat{u}) + \sum_{k=1}^{K} 6\hat{\Pi}^k(u).$$

This allows us to describe the solution of (3.5):

Proposition 5.1 *The unique optimal solution $\hat{h}$ of (3.5) is characterized as follows: for some $\hat{x}^k \in 6\hat{\Pi}^k(\hat{u}+\hat{h})$, $k = 1, \ldots, K$ and $v \in \mathrm{N}_U(\hat{u}+\hat{h})$,*

$$\hat{h} = M^{-1}\hat{g}, \quad \text{where } \hat{g} := y(\hat{u}) - \hat{x} + v,\ \hat{x} := \sum_{k=1}^{K} \hat{x}^k \in 6\hat{\Pi}(\hat{u}+\hat{h}). \tag{5.2}$$

Proof Watching for various changes of sign, apply the optimality condition [11, Theorem VII.1.1.1(iii)]: there is some supergradient in $6\hat{\theta}(\hat{u}+\hat{h})$ lying in $\mathrm{N}_U(\hat{u}+\hat{h})$. In view of the above-mentioned calculus rules, this writes

$$-y(\hat{u}) + M\hat{h} + \sum_{k=1}^{K} \hat{x}^k = v,$$

which is just (5.2). □

With the particular form of U, the property $v \in \mathrm{N}_U(\hat{u}+\hat{h})$ means that $v \le 0$ is in complementarity with $\hat{u} + \hat{h} - 1/c \ge 0$.

Note that each $\hat{x}^k$ is a convex combination as described in (5.1). To make up $\hat{x}$, one needs K sets of convex multipliers $\alpha^k \in \Delta^S$, which indeed are the KKT multipliers (not necessarily unique) associated with the constraint involving π^k in (3.5); any reasonable QP solver computes them, in addition to the optimal $(\hat{h}, \hat{\pi})$. In the next statement, this remark could also be used for an alternative proof, based on complementarity slackness:

Lemma 5.2 *With the notation of Proposition 5.1, $\hat{\Pi}^k(u) \le u^\top \hat{x}^k$ for all $u \in \mathbb{R}^n$. Equality holds for $u = \hat{u} + \hat{h}$. In particular, $\hat{\Pi}(\hat{u}+\hat{h}) = (\hat{u}+\hat{h})^\top \hat{x}$.*

Proof Apply (5.1) with $u = \hat{u} + \hat{h}$: $\hat{x}^k = \sum_s \alpha^s x^k(u^s)$ for some $\alpha \in \Delta^S$. The required inequality is therefore clear from the definition (3.2) of $\hat{\Pi}^k$. Besides, this convex combination involves only indices s such that $(\hat{u}+\hat{h})^\top x^k(s) = \hat{\Pi}^k(\hat{u}+\hat{h})$, so the stated equality holds as well; and the last statement follows by summation over k. □

The whole business of dual convergence will be to drive δ to 0, and this has interesting consequences:

Proposition 5.3 *With the notation of Proposition 5.1, $\delta = \delta_h + \delta_x + \delta_v$, where*

$$\delta_h := -\frac{1}{2}\hat{h}^\top M\hat{h} \ge 0, \qquad \delta_x := \hat{u}^\top \hat{x} - \Pi(\hat{u}) \ge 0, \qquad \delta_v := \hat{h}^\top v \ge 0. \tag{5.3}$$

Besides, for all $u \in U$,

$$\theta(u) \le \theta(\hat{u}) + \delta_x + \delta_v - (u - \hat{u})^\top \hat{g}. \tag{5.4}$$

Proof Write the definition (4.1) of δ, using (3.4) and Lemma 5.2:

$$\begin{aligned}\delta &= \Phi(\hat{u}) - \hat{h}^\top y(\hat{u}) + \frac{1}{2}\hat{h}^\top M\hat{h} + (\hat{u}+\hat{h})^\top \hat{x} - \Phi(\hat{u}) - \Pi(\hat{u}) \\ &= \frac{1}{2}\hat{h}^\top M\hat{h} + [\hat{u}^\top \hat{x} - \Pi(\hat{u})] - \hat{h}^\top (y(\hat{u}) - \hat{x})\end{aligned}$$

and (5.3) follows because $y(\hat{u}) - \hat{x} = M\hat{h} - v$ from (5.2).

Then remember from (3.3) that M is negative semi-definite: $\delta_h \ge 0$. The property $\delta_x \ge 0$ comes from Lemma 5.2; and $\delta_v = (\hat{u} + \hat{h} - \hat{u})^\top v$ is nonnegative because $v \in \mathrm{N}_U(\hat{u} + \hat{h})$.

Now take an arbitrary $u \in U$. Using feasibility of $\hat{x}^k$ in (2.2), concavity of Φ and definition of normal cones,

$$\begin{aligned}\Pi(u) &\le u^\top \hat{x} = \hat{u}^\top \hat{x} + (u - \hat{u})^\top \hat{x}, \\ \Phi(u) &\le \Phi(\hat{u}) - (u - \hat{u})^\top y(\hat{u}), \\ 0 &\le (\hat{u} + \hat{h} - u)^\top v = (\hat{u} - u)^\top v + \hat{h}^\top v.\end{aligned}$$

Summing up and disclosing appropriate δ-values:

$$\theta(u) \le \Pi(\hat{u}) + \delta_x + \Phi(\hat{u}) + (u - \hat{u})^\top(-\hat{g}) + \delta_v,$$

which is just (5.4). □

Thus, when δ is small, δ_h, δ_x and δ_v are small. If M behaves itself, $\|\hat{g}\|$ is also small and (5.4) shows that $\hat{u}$ is approximately optimal in (3.1).

6 The algorithm

We are now in a position to state the algorithm. Knowing the expression of the Klein-rock function, it works with the help of the "oracle" solving (2.2) for given $u \ge 1/c$. It uses the improvement parameters κ and κ' satisfying $0 < \kappa' \le \kappa < 1$, and the stopping tolerance $\underline{\delta} \ge 0$. The starting point $u^1 \in U$ is given, as well as the initial shortest paths $x^k(u^1)$ forming the initial bundle, and the initial quadratic model $\tilde{\Phi}$.

Algorithm 6.1 (Combined Newton-cutting-plane algorithm (`Ncp`)) Initialize $S = 1$, $\hat{u} = \hat{u}^1 = u^1$.

STEP 1 (*Trial point finding*). Find $\hat{h}$, $\hat{x} = \hat{x}^S$ and $\hat{g} = \hat{g}^S$ as described by Proposition 5.1. Compute $\delta = \delta^S$ by (4.1)

STEP 2 (*Stopping test*). If $\delta^S \le \underline{\delta}$ stop, returning $\hat{u}$ and $\hat{x}$.

STEP 3 (*Line-search*). Set $t = 1$.

STEP 3.1 (*Oracle call*). Set $u^+ := \hat{u} + t\hat{h}$. Compute $x^k(u^+)$ from (2.2) and the resulting values $\Pi(u^+), \theta(u^+)$.

STEP 3.2 (*Ascent test*). If (4.2) holds, set $\hat{u}^{S+1} = u^+$; update the quadratic approximation $\tilde{\Phi}$.

Go to Step 4.

STEP 3.3 (*Null-test*). If (4.3) holds, set $\hat{u}^{S+1} = \hat{u}^S$.

Go to Step 4.

STEP 3.4 (*Interpolation*). Select a new t "well inside" the segment $]0, t[$, say in $[0.01t,\ 0.99t]$.

Go to Step 3.1.

STEP 4 (*Bundle updating and loop*). For $k = 1, \ldots, K$, append $x^k(u^+)$ obtained in Step 3.1 to the bundle. Increase S by 1 and go to Step 1.

In Step 3.4, the simplest is to divide t by 2. More sophisticated interpolation formulae can be designed, in the spirit of cubic fitting in NLP. They must however be safeguarded, so as to satisfy the "well inside" property.

Remark 6.2 (Sum of max vs. max of sum) Our approximation of Π uses K *individual* approximations $\hat{\Pi}^k$ of (3.2). Traditional bundle methods actually ignore the summation property $\Pi = \sum_k \Pi^k$: they use just one supergradient, say $\xi^s \in 6\Pi(u^s)$ for each u^s, corresponding to the *compound* linearization $u^\top \xi^s$ of $\Pi(u)$. Here, ξ^s is of course the sum of the shortest paths $x^k(u^s)$.

Storing S linearizations needs Sn elements (as opposed to the KSn elements needed here). Besides, the Sth compound quadratic problem (3.5) simplifies to

$$\begin{aligned} &\max_{u,\pi}\{\tilde{\Phi}(u) + \pi\} \\ &\text{s.t.} \quad \pi \le u^\top \sum_{k=1}^{K} x^k(u^s), \quad \text{for } s = 1, \ldots, S, \\ &\qquad u_j \ge \frac{1}{c_j}, \quad j = 1, \ldots, n, \end{aligned}$$

which has just S linking constraints.

Even with the above simplification, Algorithm 6.1 needs potentially infinite memory. However, traditional bundle methods make use of the aggregate linearization $\hat{x}$ revealed by Proposition 5.1: a "minimal" variant would approximate Π at iteration $S + 1$ by a polyhedral function made up of two pieces only, namely

$$\min\{u^\top \hat{x}^S, u^\top \xi^{S+1}\}.$$

Needless to say, these simplifications correspond to less accurate descriptions of Π, and are therefore paid by dual iterates u^+ of probably lesser quality.

7 Dual convergence

In this section, we pretend that $\underline{\delta} = 0$ in Algorithm 6.1. Then we prove that $\liminf \delta^S = 0$; this implies that the algorithm will eventually stop if $\underline{\delta} > 0$. We use the terminology introduced in Sect. 5. First we make sure that each backtracking phase terminates.

Lemma 7.1 *Assume $\kappa + \kappa' \le 1$; let $-L \le -\ell < 0$ be lower and upper bounds on the eigenvalues of M over the segment $[\hat{u}, \hat{u} + \hat{h}]$. Then (4.2) or (4.3) holds (or both) whenever $t \le \ell/L$.*

Proof Suppose that neither (4.3) nor (4.2) holds. Subtracting and using definitions:

$$\begin{aligned}\Phi(\hat{u} + t\hat{h}) &< \Phi(\hat{u}) - t[\hat{\Pi}(\hat{u} + \hat{h}) - \Pi(\hat{u})] + (\kappa + \kappa')t\delta \\ &= \Phi(\hat{u}) + t[\tilde{\Phi}(\hat{u} + \hat{h}) - \Phi(\hat{u})] + (\kappa + \kappa' - 1)t\delta \\ &\le \Phi(\hat{u}) + t[\tilde{\Phi}(\hat{u} + \hat{h}) - \Phi(\hat{u})] \\ &= \Phi(\hat{u}) + t\left[-y(\hat{u})^\top \hat{h} + \frac{1}{2}\hat{h}^\top M \hat{h}\right].\end{aligned}$$

Apply some mean-value theorem to Φ: for example, denoting by $\tilde{M}$ the Hessian of Φ at some point between $\hat{u}$ and $\hat{u} + t\hat{h}$

$$\Phi(\hat{u} + t\hat{h}) = \Phi(\hat{u}) - ty(\hat{u})^\top \hat{h} + \frac{t^2}{2}\hat{h}^\top \tilde{M} \hat{h},$$

so that

$$-ty(\hat{u})^\top \hat{h} + \frac{t^2}{2}\hat{h}^\top \tilde{M} \hat{h} < t\left[-y(\hat{u})^\top \hat{h} + \frac{1}{2}\hat{h}^\top M \hat{h}\right].$$

Divide by $t > 0$ and simplify to obtain

$$-tL\|\hat{h}\|^2 \le t\hat{h}^\top \tilde{M} \hat{h} < \hat{h}^\top M \hat{h} \le -\ell\|\hat{h}\|^2 < 0. \qquad \square$$

Remark 7.2 Thus, Step 3 cannot loop forever: if Step 3.4 produces $t_+ \le 0.99t$, say, exit to Step 4 will occur after at most $(\log \ell - \log L)/\log 0.99$ cycles.

Another consequence is that t is "not too small" when Step 3 terminates: if $t_+ \ge 0.01t$, say, this termination produces $t \ge 0.01\ell/L$.

Establishing convergence of a bundle method amounts to proving two distinct results:

- If infinitely many ascent steps are performed, the stability centers $\hat{u}$ form a maximizing sequence of θ.
- If the sequence of stability centers stops at some $\hat{u}$, then this $\hat{u}$ maximizes θ (possibly infinitely many null-steps being needed to prove this property).

These results are proved respectively in the next two sections.

7.1 Case of infinitely many ascent steps

To simplify our analysis, we will assume here that (1.1) is feasible. This guarantees the Slater property, which is a key for an appropriate primal-dual behaviour:

Lemma 7.3 *If* (1.1) *has a feasible point, then* θ *is sup-compact on* U*: for each* $z \in \mathbb{R}$*, the set of* $u \in U$ *such that* $\theta(u) \geq z$ *is (closed and) bounded. As a result,* (3.1) *has a unique solution.*

Proof Closedness classically follows from upper semi-continuity of a dual function. Let $\mathring{x}$ and $\mathring{y} = \sum_k \mathring{x}^k$ make a feasible point. Because each $\mathring{y}_j < c_j$, we can find $\varepsilon > 0$ and $B > 0$ such that

$$\text{for } j = 1, \ldots, n, \quad \mathring{y}_j \leq y_j \leq \mathring{y}_j + \varepsilon \quad \Longrightarrow \quad f_j(y_j) \leq B.$$

Take an arbitrary $u \in U \subset \mathbb{R}^n_+$ and set $y^u := \mathring{y} + \varepsilon \frac{u}{\|u\|}$. By definition of the dual function,

$$\theta(u) \leq L(\mathring{x}, y^u, u) = f(y^u) + u^\top \left(-\mathring{y} - \varepsilon \frac{u}{\|u\|} + \sum_{k=1}^{K} \mathring{x}^k \right) = f(y^u) - \varepsilon \|u\|;$$

but $0 \leq y^u_j \leq \mathring{y}_j + \varepsilon$ for each j ($0 \leq u_j \leq \|u\|$!), hence $f(y^u) \leq nB$. We have proved that $z \leq \theta(u)$ implies $z \leq nB - \varepsilon\|u\|$.

Thus, $\theta(u) \to -\infty$ when $\|u\| \to +\infty$ in U. This classically implies that (3.1) has at least one optimal solution. Finally, this solution is unique because of strict concavity: Π is concave and (3.3) shows that Φ is strictly concave. □

Sup-compactness classically eliminates the duality gap and allows the characterization of primal-dual solutions via the superdifferential of the dual function. We will recover these results in a constructive way, by establishing appropriate convergence properties of the sequences $\hat{u}$ and $(y(\hat{u}), \hat{x})$ (the latter being established in Sect. 8 below).

Theorem 7.4 *Assume that* (1.1) *has a feasible point and let Algorithm* 6.1 *generate an infinite sequence* $\mathcal{S}$ *of ascent steps. Then the subsequences* $(\delta^s)_\mathcal{S}$, $(\hat{h}^s)_\mathcal{S}$ *and* $(\hat{g}^s)_\mathcal{S}$ *tend to* 0*; and the sequence* $\hat{u}^s$ *tends to the optimal solution of* (3.1)*.*[2]

Proof The increasing sequence $\theta(\hat{u}^s)$ has a limit, not larger than the optimal value $\bar{\theta}$ of (3.1). From Lemma 7.3, the sequence $\hat{u}^s$ is bounded; let us show that the sequence $\hat{h}$ is bounded.

With the notation of Lemma 5.1, the definition of the normal cone $\mathrm{N}_U(\hat{u} + \hat{h})$ gives $v^\top(\hat{u} - \hat{u} - \hat{h}) \leq 0$, i.e. $v^\top \hat{h} \geq 0$. Using (5.2), this can be written

$$(M\hat{h} - y(\hat{u}) + \hat{x})^\top \hat{h} \geq 0.$$

[2] Note that the whole sequence $\hat{u}^s$ coincides with $(\hat{u}^s)_\mathcal{S}$, since $\hat{u}^{s+1} = \hat{u}^s$ if $s \notin \mathcal{S}$.

Now M is negative definite and (3.3) shows that its largest eigenvalue is bounded away from 0; say $\hat{h}^\top M\hat{h} \le -\varepsilon\|\hat{h}\|^2$. So we can write

$$-\varepsilon\|\hat{h}\|^2 \ge \hat{h}^\top M\hat{h} \ge (y(\hat{u}) - \hat{x})^\top \hat{h} \ge -\|y(\hat{u}) - \hat{x}\|\,\|\hat{h}\|,$$

i.e. $\|\hat{h}\| \le \|y(\hat{u}) - \hat{x}\|$. Because $y(\hat{u})$ and $\hat{x}$ are both bounded, this shows that $\hat{h}$ is bounded.

Then the eigenvalues $L > 0$ [resp. $\ell > 0$] of Lemma 7.1 are bounded from above [resp. away from 0]. Remembering Remark 7.2, t is bounded away from 0: $t \ge \underline{t} > 0$ (say with $\underline{t} = 0.01\ell/L$). Then we have from (4.2)

$$\begin{aligned} &\theta(\hat{u}^{s+1}) \ge \theta(\hat{u}^s) + \kappa\underline{t}\delta^s \quad && \text{if } s \in \mathcal{S},\\ &\theta(\hat{u}^{s+1}) = \theta(\hat{u}^s) && \text{otherwise} \end{aligned}$$

and we obtain by summation $\sum_{s\in\mathcal{S}} \delta^s \le [\bar{\theta} - \theta(u^1)]/\kappa\underline{t}$: $(\delta^s)_{\mathcal{S}}$ tends to 0. The three components of δ^s in (5.3) tend to 0 as well, and this is also true of the subsequences $\{\hat{g}^s\}_{\mathcal{S}}$ and $\{\hat{h}^s\}_{\mathcal{S}}$.

Then take a cluster point of $\hat{u}^s$ and pass to the limit in (5.4): this cluster point has to be the unique optimal solution of (3.1). □

Note that, if (1.1) has no feasible point, then $\theta(\hat{u})$ will typically tend to $+\infty$; δ has no reason to tend to 0, the stop in Algorithm 6.1 will never occur. To prevent this situation, it is wise to insert an "emergency stop" when $\theta(\hat{u})$ is unduly large.

7.2 Case of finitely many ascent steps

First we show that the next QP will modify the present $\hat{h}$ (remember from Proposition 3.1 that $\hat{\Pi}(\hat{u} + \hat{h}) = \hat{\pi}$):

Lemma 7.5 *If (4.3) holds, the new linearization $x(\hat{u} + t\hat{h})$ satisfies*

$$(\hat{u} + \hat{h})^\top x(\hat{u} + t\hat{h}) \le \hat{\Pi}(\hat{u} + \hat{h}) - \kappa'\delta. \tag{7.1}$$

Proof Use simplified notation: set $z := (\hat{u} + \hat{h})^\top x^+$, with $x^+ := x(\hat{u} + t\hat{h}) = x(u^+)$.

Because $x^+ \in 6\Pi(u^+)$, we have by definition $\Pi(\hat{u}) \le \Pi(u^+) - t\hat{h}x^+$; hence $\hat{h}^\top x^+ \le [\Pi(u^+) - \Pi(\hat{u})]/t$, so that

$$\begin{aligned} z &= \Pi(u^+) + (1-t)\hat{h}^\top x^+ \le \Pi(u^+) + \frac{1-t}{t}[\Pi(u^+) - \Pi(\hat{u})]\\ &= \frac{1}{t}\Pi(u^+) - \frac{1-t}{t}\Pi(\hat{u}). \end{aligned}$$

Now use (4.3) to bound $\Pi(u^+)$:

$$z \le \frac{1}{t}\Pi(\hat{u}) + \hat{\Pi}(\hat{u} + \hat{h}) - \Pi(\hat{u}) - \kappa'\delta - \frac{1-t}{t}\Pi(\hat{u}),$$

which is just (7.1). □

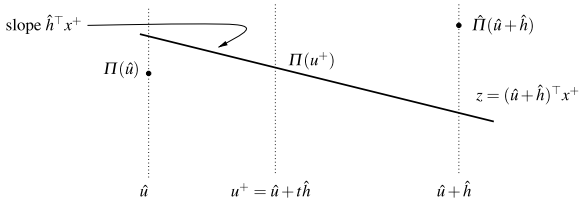

Fig. 4 The new linearization passes under $\hat{\Pi}(\hat{u}+\hat{h})$

The proof of the next result uses explicitly the fact that *all* linearizations are stored in the bundle: to accommodate the bundle compression alluded to at the end of Remark 6.2, a more sophisticated proof would be required, along the lines of [11, Theorem XV.3.2.4].

Theorem 7.6 *Suppose the stability center stops at some iteration* S: $\hat{u}^{s+1}=\hat{u}^s$ *for all* $s\geq S$. *Then* $\delta^s\to 0$ *and* $\hat{u}^S$ *is the optimal solution of* (3.1).

Proof The situation is as follows: at all iterations s following S, $\hat{u}=\hat{u}^S$, M and $y(\hat{u})$ are fixed; $\delta=\delta^s$ forms a nonincreasing sequence since (3.5) has more and more constraints; Proposition 5.3 then guarantees that $\hat{h}=\hat{h}^s$ is bounded. It follows that $u^{s+1}=\hat{u}+t^s h^s$ is also bounded, as lying in the segment $[\hat{u},\hat{u}+\hat{h}^s]$; hence $x(u^s)\in \mathfrak{b}\Pi(u^s)$ is bounded ([11, Proposition VI.6.2.2]).

Write (7.1) and the definition (3.2) of $\hat{\Pi}$ at the sth iteration: for all $s>S$ and all $r\leq s$,

$$(\hat{u}+\hat{h}^s)^\top x(u^{s+1})+\kappa'\delta^s\leq\hat{\Pi}(\hat{u}+\hat{h}^s)\leq(\hat{u}+\hat{h}^s)^\top x(u^r),$$

so that

$$\kappa'\delta^s\leq(\hat{u}+\hat{h}^s)^\top[x(u^r)-x(u^{s+1})]\leq B\|x(u^r)-x(u^{s+1})\|,$$

where we have used the Cauchy-Schwarz inequality and B is a bound for $\|\hat{u}+\hat{h}^s\|$. Now assume $\delta^s\geq\varepsilon>0$ for all s. Then

$$\|x(u^r)-x(u^{s+1})\|\geq\frac{\kappa'\varepsilon}{B}\quad\text{for all } s>S \text{ and all } r\leq s.$$

In words: around each $x(u^r)$, there is a ball of fixed radius $\kappa'\varepsilon/B$ which cannot contain any other x; because the x's are confined in a bounded set, this is impossible.

It follows that the monotone sequence δ^s tends to 0, pass to the limit in (5.4) to establish optimality of $\hat{u}$. □

8 Primal recovery

It is known that convergence of the dual algorithm has its counterpart concerning the primal problem. However, we solve here (3.1), while the dual of (1.1) is rather (2.3). The issue is therefore more delicate, especially when infinitely many ascent steps are performed; we analyze this case first.

Theorem 8.1 *Make the assumptions of Theorem 7.4. Then*:

- *The subsequence* $\{y(u^s)\}_{s\in\mathcal{S}}$ *tends to the unique* y*-optimal solution of* (1.1);
- *for* $k=1,\ldots,K$, *the subsequences* $\{\hat{x}^{k,s}\}_{s\in\mathcal{S}}$ *are bounded and any of their cluster points makes up an* x*-optimal solution of* (1.1).

Proof We already know from Theorem 7.4 that $\{\delta^s\}_{\mathcal{S}}$, $\{\hat{h}^s\}_{\mathcal{S}}$ and $\{\hat{g}^s\}_{\mathcal{S}}$ tend to 0. We also know that $\hat{u}^s$ has a limit $\bar{u}$; therefore $y(\hat{u}^s)\to y(\bar{u})$ and we proceed to prove that $\{\hat{x}^s\}_{\mathcal{S}}\to y(\bar{u})$. Note that $\{\hat{u}^s+\hat{h}^s\}_{\mathcal{S}}\to\bar{u}$.

Define the set $J^*:=\{j=1,\ldots,n:\bar{u}_j=1/c_j\}$ of artificial constraints that are active at $\bar{u}$.

- For $j\notin J^*$, $\bar{u}_j>1/c_j$ so that $\hat{u}^s_j+\hat{h}^j_j>1/c_j$ for $s\in\mathcal{S}$ large enough. The property $v^s\in \mathrm{N}_U(\hat{u}^s+\hat{h}^s)$ therefore implies $v^s_j=0$, hence $\hat{x}^s_j=y_j(\hat{u}^s)-\hat{g}^s_j$ tends to $y_j(\bar{u})$.
- For $j\in J^*$, $y_j(\bar{u})=0$; hence $y_j(\hat{u}^s)\to 0$ and $\hat{x}^s_j\to 0$ because, from (5.2),

$$0\le\hat{x}^s_j=y_j(\hat{u}^s)-\hat{g}^s_j+v^s_j\le y_j(\hat{u}^s)-\hat{g}^s_j\to 0.$$

Piecing together, we see that

$$\{\hat{x}^s-y(\hat{u}^s)\}_{\mathcal{S}}\to 0. \tag{8.1}$$

Now write

$$\theta(\hat{u}^s)=\Phi(\hat{u}^s)+\Pi(\hat{u}^s)=f(y(\hat{u}^s))-(\hat{u}^s)^\top y(\hat{u}^s)+\Pi(\hat{u}^s)$$

and pass to the limit for $u\in\mathcal{S}$:

$$\theta(\bar{u})=f(y(\bar{u}))-\bar{u}^\top y(\bar{u})+\Pi(\bar{u}).$$

But observe from (8.1) that

$$-\bar{u}^\top y(\bar{u})+\Pi(\bar{u})=\lim_{s\in\mathcal{S}}[-(\hat{u}^s)^\top\hat{x}^s+\Pi(\hat{u}^s)]=\lim_{s\in\mathcal{S}}\delta^s_x$$

where we have used the notation of Proposition 5.3. Since $\{\delta^s_x\}_{\mathcal{S}}\to 0$,

$$\theta(\bar{u})=f(y(\bar{u})). \tag{8.2}$$

Finally, we know that the subsequences $\{\hat{x}^{k,s}\}_{\mathcal{S}}$ are bounded. Consider a cluster point: say, with $\mathcal{S}'\subset\mathcal{S}$, $\{\hat{x}^{k,s}\}_{\mathcal{S}'}\to\bar{x}^k$ for $k=1,\ldots,K$. The $\bar{x}^k$'s are feasible in (1.1) and they sum up to $y(\bar{u})$: $(\bar{x},y(\bar{u}))$ makes a feasible point in (1.1). In view of (8.2), weak duality tells us that this point is primal optimal. □

The case of finitely many ascent steps is just easier, as $\hat{u}^s$ reaches its limit $\bar{u}$ for some finite s.

Theorem 8.2 *Suppose that the stability center stops at some iteration* S. *Then the conclusions of Theorem 8.1 hold, with* $\mathcal{S}$ *replaced by the whole sequence* $S+1,\ldots$. *In fact,* $\hat{u}^S$ *is the optimal solution of* (3.1).

Proof Invoke Theorem 7.6: the whole sequences δ^s, $\hat{h}^s$ and $\hat{g}^s$ converge to 0. Then proceed exactly as for Theorem 8.1, with the simplifying property that $\hat{u}^s = \bar{u}$ for all $s > S$. □

Note that this result makes no assumption about primal feasibility ... and yet proves primal existence! This has an interesting consequence:

Corollary 8.3 *Suppose that* (1.1) *has no feasible point. Then the dual function* θ *does not reach its maximum.*

Proof Suppose for contradiction that (3.1) has an optimal solution $\bar{u}$. Initialize Algorithm 6.1 with $u^1 = \bar{u}$. There can be no descent step and Theorem 8.2 establishes the existence of an optimal primal solution. □

9 Numerical illustrations

To get a feeling of the numerical merits of our approach, we benchmark it on 16 test-problems against two implementations of its direct concurrent, namely the standard bundle method, which we briefly recall now.

If no attention is paid to its smoothness, Φ can be approximated via the linearizations $\bar{\Phi}_j^s(u_j) := \Phi_j(u_j) - (u_j - u_j^s)^\top y_j(u_j^s)$, instead of the quadratic functions $\tilde{\Phi}_j(u_j)$ of (3.4). Then Φ can be approximated just as Π by a polyhedral function (call it $\hat{\Phi}$) instead of the quadratic function $\tilde{\Phi}$ of (3.4); then a bundle method maximizes the resulting polyhedral approximation $\hat{\Phi} + \hat{\Pi}$ of θ, stabilized by a quadratic term $\frac{1}{2t}\|u - \hat{u}\|^2$; $t > 0$ is a parameter. Standard bundle methods maximize this approximation, and then manage the stability center $\hat{u}$ just as in Algorithm 6.1 (except that no backtracking is necessary).

An implementation in the spirit of the present paper uses the *individual* approximations $\Phi_j(u_j) \le \hat{\Phi}_j(u_j) := \min_s \bar{\Phi}_j^s(u_j)$, thus replacing (3.5) by

$$\max_{h,\phi,\pi} \left\{ \sum_{j=1}^n \phi_j + \sum_{k=1}^K \pi^k - \frac{1}{2tS}\|h\|^2 \right\}$$
$$\text{s.t.} \quad \left.\begin{array}{ll} \phi_j \le \bar{\Phi}_j^s(\hat{u}_j + h_j), & j = 1,\dots,n, \\ \pi^k \le (\hat{u} + h)^\top x^k(u^s), & k = 1,\dots,K, \end{array}\right\} \; s = 1,\dots,S,$$
$$h_j \ge \frac{1}{c_j} - \hat{u}_j, \quad j = 1,\dots,n.$$

We will refer to this implementation as `bfull`, as it fully splits the approximation of θ. Now remember Remark 5: ignoring the summation in Φ, we can also use the *compound* (less accurate) polyhedral approximation $\Phi(u) \le \min_s \sum_j \bar{\Phi}_j^s(u_j)$. In

compensation, the resulting quadratic program simplifies to

$$\max_{h,\phi,\pi}\left\{\phi+\sum_{k=1}^{K}\pi^k-\frac{1}{2t^S}\|h\|^2\right\}$$
$$\left.\begin{array}{ll}\text{s.t.} & \phi\le\displaystyle\sum_{j=1}^{n}\bar{\Phi}_j^s(\hat{u}_j+h_j),\\ & \pi^k\le(\hat{u}+h)^\top x^k(u^s),\quad k=1,\dots,K,\\ & h_j\ge\dfrac{1}{c_j}-\hat{u}_j,\quad j=1,\dots,n.\end{array}\right\}\ s=1,\dots,S,$$

We also compare our method to this implementation, referred to as `bhalf`: it uses only a half of the splitting possibilities in θ.

With Algorithm 6.1 (referred to as `Ncp`), this makes three solvers, which have been implemented in C on a bi-processor Intel Xeon (2.4 GHz, 1.5 GB RAM) under Linux operating system. Both standard bundle variants are home-made implementations of [14] (in particular for the t-management); Dijkstra's algorithm is used for the shortest path problems and the various QP are solved by Cplex 10.0. We use $\kappa=\kappa'=0.1$ in Algorithm 6.1; the stopping criterion is $\underline{\delta}=10^{-6}$ for `Ncp` and $\varepsilon=10^{-6}$ for `bfull` and `bhalf`. We group the commodities by source nodes so that each computation of Π calls at most m times Dijkstra's algorithm.

The results are summarized in Table 1.

- The first group of 4 columns describes the 16 test problems, ranked by number K of commodities (recall that the number of dual variables is n). Problems 3, 7, 10 and 13 are the random networks already used in [24]. Problems 1 and 4 are nso22h and nso148 of [9], well known in the convex optimization community. The remaining test problems are based on actual networks.
- The next group of columns gives the number of QP solved. This is also the number of oracle calls for both variants of standard bundle; for `Ncp`, add the number of backtracks (given in parenthesis).
- Computing times are in seconds; they are mainly indicative and would probably change substantially with the use of a specialized QP solver such as [4, 13, 15].
- For each test-problem, `Ncp` obtains the best θ-value, which is given in the last column. The previous two columns give the final relative gap obtained by the two variants of standard bundle.

This table is rather eloquent. In terms of accuracy, all methods are comparable, except three failures of `bhalf`; but `Ncp` is drastically faster. First, it always requires less iterations. Also, the work per iteration is definitely smaller for `bhalf`, which solves a much cheaper quadratic program. Nevertheless, its computing time is overwhelmed by the two others', even forgetting the three instances where the stopping criterion could not be reached.

If comments on the behaviour of `Ncp` should be ventured, we could observe that the number of QP resolutions is remarkably and consistently small. This means that the polyhedral part Π of θ is easily approximated by the polyhedral model $\hat{\Pi}$ (only

Table 1 Comparison of Algorithm 6.1 (`Ncp`) with two alternative standard bundle implementations (`bfull` and `bhalf`). Number of backtracks for `Ncp` is given in parenthesis

Pb	m	n	K	Iterations			Total CPU			Final accuracy w.r.t. `Ncp`		best θ (from `Ncp`)
				`Ncp`	`bfull`	`bhalf`	`Ncp`	`bfull`	`bhalf`	`bfull`	`bhalf`	
1	14	22	23	12 (88)	19	645	0.03	0.20	286.2	0.	10^{-6}	103.412019
2	19	68	30	5 (6)	11	16	0.02	0.14	0.08	10^{-7}	0.	8.994992
3	60	280	100	7 (95)	14	93	0.20	1.47	8.01	0.	0.	53.080767
4	61	148	122	7 (8)	24	167	1.07	8.84	61.09	0.	2×10^{-8}	151.926869
5	20	64	133	7 (8)	16	156	0.35	2.67	70.59	0.	5×10^{-8}	39.635462
6	122	332	162	9 (90)	21	309	0.61	5.23	495.4	0.	10^{-8}	276.321357
7	100	600	200	7 (96)	17	190	0.78	6.91	250.6	10^{-8}	2×10^{-8}	84.967483
8	30	72	335	7 (104)	24	1000[a]	0.13	3.56	5982.1	10^{-7}	3×10^{-6}	36.451716
9	21	68	420	7 (8)	42	151	5.62	88.66	856.5	0.	6×10^{-7}	68.838958
10	100	800	500	10 (304)	16	274	9.60	26.32	2061	4×10^{-8}	8×10^{-7}	139.096514
11	67	170	761	8 (133)	32	158	4.84	64.26	409.	3×10^{-8}	5×10^{-8}	109.895582
12	34	160	946	5 (6)	14	285	1.03	12.20	1650.2	5×10^{-8}	5×10^{-8}	19.566682
13	300	2000	1000	11 (319)	22	569	73.37	322.51	30564.	7×10^{-9}	2×10^{-7}	304.389462
14	48	198	1583	9 (80)	22	803	13.09	68.45	5 h 45	10^{-7}	4×10^{-7}	135.463182
15	81	188	2310	3 (4)	19	1000[a]	2.44	308.51	28 h	3×10^{-7}	9×10^{-4}	41.791840
16	122	342	2881	9 (73)	30	1000[a]	311.85	764.5	42 h	2×10^{-7}	2×10^{-2}	242.714771

[a]Failures to obtaining required accuracy