Optim Lett
DOI 10.1007/s11590-017-1168-z

ORIGINAL PAPER

Dynamic smoothness parameter for fast gradient methods

Antonio Frangioni[1] · **Bernard Gendron**[2,3] · **Enrico Gorgone**[4,5]

Received: 6 March 2017 / Accepted: 14 July 2017

Abstract We present and computationally evaluate a variant of the fast gradient method by Nesterov that is capable of exploiting information, even if approximate, about the optimal value of the problem. This information is available in some applications, among which the computation of bounds for hard integer programs. We show that dynamically changing the smoothness parameter of the algorithm using this information results in a better convergence profile of the algorithm in practice.

Keywords Fast gradient method · Lagrangian relaxation · Convex optimization

✉ Enrico Gorgone
egorgone@unica.it

Antonio Frangioni
frangio@di.unipi.it

Bernard Gendron
Bernard.Gendron@cirrelt.ca

1 Dipartimento di Informatica, Università di Pisa, Pisa, Italy

2 Centre Interuniversitaire de Recherche sur les Réseaux d'Entreprise, la Logistique et le Transport (CIRRELT), Montreal, Canada

3 Department of Computer Science and Operations Research, Université de Montréal, Montreal, Canada

4 Indian Institute of Management Bangalore (IIMB), Bangalore, India

5 Dipartimento di Matematica e Informatica, Università di Cagliari, Cagliari, Italy

Published online: 20 July 2017

1 Introduction

One of the crucial components of solution algorithms for mixed integer linear programs (MILP) is the computation of tight bounds upon the optimal value of the problem. Although the solution of the continuous relaxation (CR) of the MILP, usually strengthened by valid inequalities, is often the method of choice, forming a Lagrangian relaxation (LR) and (approximately) solving the corresponding Lagrangian dual (LD) can be preferable in some cases. This is true in particular when the LR decomposes into several smaller subproblems (e.g., [8,9] and the references therein). The LD is typically a non-smooth problem, and it is usually solved by algorithms of two different families: *subgradient methods* (SM) [6,9,14] and *bundle methods* (BM) [7,8,10]. The former are easier to implement and their iteration cost is dominated by the function computation, whereas the latter are more complex and require the solution of a (potentially, costly) subproblem at each iteration; however, they have better convergence in practice. The right trade-off depends on many factors, among which the required (relative or absolute) accuracy; the numerical experiments of [9] show that SM can be competitive, in a prototypical application, provided that a substantial amount of tuning is performed to choose the many algorithmic parameters. Among SM, the primal-dual variants (PDSM) [12] are particularly attractive because they have much fewer parameters to tune. However, their practical performance might be worse than that of other variants. The analysis in [9] seems to indicate that one of the factors at play is that most SM, but not PDSM, can incorporate external information about the optimal value of the problem (in particular, for the selection of the stepsize). Hence, exploiting this information might be useful computationally.

This work provides an initial step towards that goal by analyzing a different, but related, family of non-smooth optimization algorithms, that of *fast gradient* methods (FG) [1–3,11,13], that have efficiency estimates of the order $O(1/\epsilon)$—with ϵ the required absolute accuracy—whereas the complexity of any black-box non-smooth method is at best $O(1/\epsilon^2)$. The downside is that FG require an explicit modification of the oracle, which might negatively impact the total running time. In the standard version, FG do not exploit any knowledge on the optimal value. However they have one crucial *smoothness parameter* that is naturally related with the current distance (on the value axis) from the optimum. We propose a simple scheme, in two variants, for dynamically managing the smoothness parameter to exploit (approximate) information on the optimal value, showing that this leads to a significant improvement of the convergence profile of the approach. We test the variant on two different LD of a hard MILP. The approach could be useful in several other applications particularly suited to FG, such as imaging [1,4].

2 The method

We study approaches for the numerical solution of the problem

$$f_* = \min\left\{ f(\lambda) = \hat{f}(\lambda) + \max\{\langle B\lambda, z\rangle - \phi(z) : z \in Z\} : \lambda \in \Lambda \right\} \tag{1}$$

where $\Lambda \subseteq \mathbb{R}^n$ is closed and convex, and $f : \mathbb{R}^n \to \mathbb{R}$ is a proper convex nondifferentiable function due to the inner maximization (being ϕ continuous and convex on

the bounded closed convex set Z and B a linear operator), while $\hat{f} \in C^{1,1}$. The idea of FG methods is to make (1) smooth by defining

$$f_\mu(\lambda) = \hat{f}(\lambda) + \max\{\langle B\lambda, z\rangle - \phi(z) - \mu r_2(z) : z \in Z\}, \tag{2}$$

which is a smooth lower approximation of f if the *prox-function* $r_2(z) \geq 0$ is continuous and strongly convex on Z. The *smoothness parameter* $\mu > 0$ connects the minima of f and f_μ, so appropriately managing μ one can apply a fast gradient approach to f_μ and obtain an approximate solution to (1). This approach has been successfully applied in machine learning, data mining, inverse problems, and imaging [1,4], and has inspired further research [2,3,11].

The FG is based on two *prox-functions*, that for simplicity we take as $r_1(\lambda) = \|\lambda - \bar{\lambda}\|^2/2$ and $r_2(z) = \|z - \bar{z}\|^2/2$, $\bar{\lambda}$ and $\bar{z}$ being the centers. Since Z is bounded, $\max\{r_2(z) : z \in Z\} \leq R_2 < \infty$; therefore, $f_\mu(\lambda) \leq f(\lambda) \leq f_\mu(\lambda) + \mu R_2$, which implies that any method minimizing f_μ over Λ leads to an approximate solution of (1) if $\mu \searrow 0$. Given the (unique) optimal solution $z^*_\mu(\lambda)$ of (2), $\nabla f_\mu(\lambda_k) = \nabla \hat{f}(\lambda_k) + z^*_\mu(\lambda_k)B$; it can be seen [13, Theorem 1] that ∇f_μ is Lipschitz continuous with constant $L_\mu = M + \|B\|^2/\mu$, where M is the Lipschitz constant of $\nabla \hat{f}$. For any μ, the FG approach to minimizing f_μ is based on arbitrarily selecting a sequence of weights υ_k such that $\upsilon_0 \in (0, 1]$ and $\upsilon_k^2 \leq \Delta_k = \sum_{i=0}^k \upsilon_i$ for $k \geq 1$, and solving the two problems

$$\pi_k = \arg\min\left\{\langle \nabla f_\mu(\lambda_k), \lambda - \lambda_k\rangle + L_\mu\|\lambda - \lambda_k\|^2/2 : \lambda \in \Lambda\right\} \tag{3}$$

$$\zeta_k = \arg\min\left\{L_\mu r_1(\lambda) + \sum_{i=0}^k \upsilon_i\left[f_\mu(\lambda_i) + \langle \nabla f_\mu(\lambda_i), \lambda - \lambda_i\rangle\right] : \lambda \in \Lambda\right\} \tag{4}$$

Then, with $\iota_{k+1} = \upsilon_{k+1}/\Delta_{k+1}$, the next iterate is computed as $\lambda_{k+1} = \iota_{k+1}\zeta_k + (1 - \iota_{k+1})\pi_k$ (with $\lambda_0 = \bar{\lambda}$). We now reproduce the convergence analysis of [13] replacing the requirement that Λ is bounded, which does not hold in our application, with $f_* = f(\lambda^*) > -\infty$, so that $R_1 = r_1(\lambda^*) < \infty$. As in the original development we take $\upsilon_k = (k + 1)/2$, so that $\Delta_k = (k + 1)(k + 2)/4$.

Proposition 1 *Under the assumptions (i) $f_* = f(\lambda^*) > -\infty$, (ii) $R_1 < \infty$ and (iii) $M = 0$, for any $\epsilon > 0$ by setting $\mu = \epsilon/(2R_2)$ the inequality $f(\pi_k) - f_* \leq \epsilon$ is satisfied in at most $k + 1 = 4\|B\|\sqrt{R_1 R_2}/\epsilon$ iterations.*

Proof By [13, Theorem 2], for any $k \geq 0$ we have

$$\Delta_k f_\mu(\pi_k) \leq \min\left\{L_\mu r_1(\lambda) + \sum_{i=0}^k \upsilon_i[f_\mu(\lambda_i) + \langle \nabla f_\mu(\lambda_i), \lambda - \lambda_i\rangle] : \lambda \in \Lambda\right\},$$

and from both convexity and $\Delta_k = \sum_{i=0}^k \upsilon_i$ it follows that

$$\Delta_k f_\mu(\pi_k) \leq \min\left\{L_\mu r_1(\lambda) + \sum_{i=0}^k \upsilon_i f_\mu(\lambda) : \lambda \in \Lambda\right\} \leq L_\mu R_1 + \Delta_k f_\mu(\lambda^*).$$

Using $L_\mu = M + \|B\|^2/\mu$ we get $\Delta_k f_\mu(\pi_k) \leq (M + \|B\|^2/\mu)R_1 + \Delta_k f_\mu(\lambda^*)$, and therefore $f_\mu(\pi_k) - f_\mu(\lambda^*) \leq (1/\Delta_k)\big(M + \|B\|^2/\mu\big)R_1$. The fact that $f_\mu \leq f$ implies

that $f_\mu(\lambda^*) \leq f_*$. In addition, $f(\lambda) \leq f_\mu(\lambda) + \mu R_2$ holds for any λ and, hence, in particular for π_k, yielding

$$f(\pi_k) - f_* \leq (1/\Delta_k) \left(M + \|B\|^2/\mu \right) R_1 + \mu R_2.$$

One can then use $\Delta_k = (k+1)(k+2)/4$ and find the value of μ minimizing the right-hand side above; this gives $\mu = (2\|B\|\sqrt{R_1/R_2})/(k+1)$, whence

$$0 \leq f(\pi_k) - f_* \leq 4 \left(M R_1/(k+1) + \|B\|\sqrt{R_1 R_2} \right) \big/ (k+1) \leq \epsilon$$

from which the desired result immediately follows. □

The minimization problems (3)–(4) actually reduce to closed-form formulæ when either $\Lambda = \mathbb{R}^n$ or $\Lambda = \mathbb{R}^n_+$. Indeed, in the first case $\pi_k = \bar{\pi}_k = \lambda_k - \nabla f_\mu(\lambda_k)/L_\mu$ and $\zeta_k = \bar{\zeta}_k = \bar{\lambda} - \sum_{i=0}^{k-1} \upsilon_i \nabla f_\mu(\lambda_i)/L_\mu$, while in the second case $\pi_k = \max\{0, \bar{\pi}_k\}$ and $\zeta_k = \max\{0, \bar{\zeta}_k\}$. Furthermore, the simple recursive formula $d_k = \iota_k \nabla f_\mu(\lambda_k) + (1 - \iota_k) d_{k-1} = (1/\Delta_k) \sum_{i=0}^{k} \upsilon_i \nabla f_\mu(\lambda_i)$, whose correctness is easily verified by induction, can be used to avoid keeping all the gradients to compute ζ_k, thereby making each iteration inexpensive.

The analysis therefore suggests to keep μ fixed to a value directly proportional to the desired *absolute* error ϵ. Because typically one wants to specify *relative* tolerances ϵ_r instead, the practical implementation must be akin to

$$\mu = \epsilon_r |f_{ref}| / (2R_2) \tag{5}$$

where f_{ref} is some reference value providing an estimate of f_*. In some applications a lower bound $f_{lb} \leq f_*$ is available that can be used as f_{ref}. However, knowledge of f_{lb} could be put to even better use. Indeed, μ is proportional to ϵ, and the algorithm basically performs steps of $1/L_\mu = \mu/\|B\|^2$ (if $M = 0$) along the direction d_k, as recalled above. Therefore, a small value of μ, necessary to attain a high accuracy, leads to small steps when one if "far" from f_*. It would therefore be intuitively attractive to have larger values of μ early on and reduce it as the algorithm proceeds. Availability of f_{lb} suggests the rule

$$\mu_k = \max \left\{ f_k^{best} - f_{lb}, \epsilon_r |f_{lb}| \right\} \big/ (2R_2), \tag{6}$$

where $f_k^{best} = \min\{f(\lambda_i) : i \leq k\}$. It is clear that such a modification still yields a convergent algorithms. Indeed, one could choose a finite sequence $\{\epsilon_i\} \to \epsilon$ and iteratively run the algorithm with fixed ϵ_i until that accuracy is attained, then move to the next value; this is obviously still convergent. Knowledge of f_{lb} just allows to change ϵ_i at every iteration rather than waiting for the number of iterations estimated by Proposition 1. In the next section we show that (6) actually improves the convergence rate of the algorithm when f_{lb} is accurate, and can be modified to handle the case when it is not.

3 Application to multicommodity network design

The fixed-charge multicommodity capacitated network design problem (FC-MCND) is a general network design problem with many applications (see [5,8,9] and the references therein). Efficiently computing tight lower bounds on its optimal value is crucial for solution approaches, and Lagrangian techniques have been shown to be competitive. In [9], gradient-like approaches have been thoroughly analysed, showing how the availability of lower bounds on the optimal value improves the efficiency of solution approaches that can make use of this information. We aim at verifying if an analogous phenomenon occurs for FG, that can also be applied to FC-MCND as briefly described here. The data of FC-MCND is a directed graph $G = (N, A)$, where F_i and B_i respectively denote the set of outbound and inbound arcs of node $i \in N$, and a set of commodities K. Each $k \in K$ has a *deficit vector* $b^k = [b_i^k]_{i \in N}$ that denotes the net amount of flow asked at each node. Each arc $(a_+, a_-) = a \in A$ can only be used if the corresponding *fixed cost* $f_a > 0$ is paid, in which case the *mutual capacity* $u_a > 0$ bounds the total amount of flow on a, while *individual capacities* u_a^k bound the flow of commodity k. The *routing cost* c_a^k has to be paid for each unit of commodity k moving through a. A formulation is

$$\min \sum_{k \in K} \sum_{a \in A} c_a^k x_a^k + \sum_{a \in A} f_a y_a \tag{7}$$

$$\sum_{a \in F_i} x_a^k - \sum_{a \in B_i} x_a^k = b_i^k \qquad i \in N, k \in K \tag{8}$$

$$\sum_{k \in K} x_a^k \leq u_a y_a \qquad a \in A \tag{9}$$

$$x_a^k \leq u_a^k y_a \qquad a \in A, k \in K \tag{10}$$

$$0 \leq x_a^k \leq u_a^k \qquad a \in A, k \in K \tag{11}$$

$$y_a \in \{0, 1\} \qquad a \in A \tag{12}$$

Two classical approaches for deriving lower bounds on its optimal value are the *flow relaxation* (FR) and the *knapsack relaxation* (KR). In the former one relaxes constraints (9)–(10) with multipliers $\lambda = [\alpha, \beta] = [\alpha_a, \beta_a^k]_{a \in A, k \in K} \geq 0$. This yields the objective function

$$\min \sum_{k \in K} \sum_{a \in A} \left(c_a^k + \alpha_{ij} + \beta_a^k\right) x_a^k + \sum_{a \in A} \left(f_a - \alpha_a u_a - \sum_{k \in K} u_a^k \beta_a^k\right) y_a$$

whose minimization subject to the remaining (8), (11)–(12) reduce to $|K|$ single-commodity linear minimum cost network (MCF) problems plus $|A|$ trivial single-variable integer problems. Applying FG means adding to (7) the term

$$\mu \sum_{a \in A} \left[(y_a - \bar{y}_a)^2 + \sum_{k \in K} (x_a^k - \bar{x}_a^k)^2 \right] \Big/ 2 \tag{13}$$

with arbitrary $\bar{x}$ and $\bar{y}$, yielding $f_\mu(\lambda) = f^0 + \sum_{k \in K} f_\mu^k(\lambda) + \sum_{a \in A} f_\mu^a(\lambda)$ with

$$f^0 = -\sum_{a\in A} \mu \left[(\bar{y}_a)^2 + \sum_{k\in K} (\bar{x}_a^k)^2 \right] \Big/ 2$$

$$f_\mu^k(\lambda) = -\min \left\{ \sum_{a\in A} \left[\bar{c}_a^k x_a^k + \mu (x_a^k)^2/2 \right] : (8), (11) \right\} \tag{14}$$

$$f_\mu^a(\lambda) = -\min \left\{ \bar{f}_a y_a + \mu y_a^2/2 : (12) \right\} \tag{15}$$

where $\bar{c}_a^k = c_a^k + \alpha_a + \beta_a^k - \mu \bar{x}_a^k$ and $\bar{f}_a = f_a - \alpha_a u_a - \sum_{k\in K} u_a^k \beta_a^k - \mu \bar{y}_a$; (14) is now a (convex, separable) *quadratic* MCF problem, which is still efficiently solvable, albeit less so in practice than the linear version. In order to apply FG the R_2 constant has to be computed by maximizing (13) over (8), (11)–(12), which is a hard problem. Yet it decomposes in $|K| + |A|$ independent subproblems, the latter being single-variable ones. For the remaining part we use the linear upper approximation of $(x_a^k - \bar{x}_a^k)^2$ given by the gradient computed at $x_a = u_a^k/2$, i.e., $R_2 \le (\sum_{k\in K} R_2^k + \sum_{a\in A} \max\{\bar{y}_a^2, (1 - \bar{y}_a)^2\})/2$ with

$$R_2^k = \sum_{a\in A} (\bar{x}_a^k)^2 + \max \left\{ \sum_{a\in A} \left(u_a^k/2 - \bar{x}_a^k \right) x_a^k : (8), (11) \right\}.$$

In the KR, one rather dualizes the flow conservation constraints (8) with multipliers $\lambda = [\lambda_i^k]_{i\in N, k\in K}$; this yields the objective function

$$\min \sum_{a\in A} \left[\sum_{k\in K} \left(c_a^k + \lambda_{a_+}^k - \lambda_{a_-}^k \right) x_a^k + f_a y_a \right] + \sum_{i\in N} \sum_{k\in K} \lambda_i^k b_i^k$$

whose minimization subject to (9)–(12) reduce to $|A|$ independent continuous knapsack problems (KP). Applying FG corresponds again to adding (13), leading to $f_\mu(\lambda) = f^0 + \sum_{a\in A} f_\mu^a(\lambda)$ with

$$f^0 = -\sum_{i\in N} \sum_{k\in K} \lambda_i^k b_i^k - \mu \sum_{a\in A} \left(\bar{y}_a^2 + \sum_{k\in K} (\bar{x}_a^k)^2 \right) \Big/ 2$$

$$f_\mu^a(\lambda) = -\min \left\{ \left(g^a(\lambda) + f_a - \mu \bar{y}_a \right) y_a : (12) \right\}$$

$$g^a(\lambda) = \min \left\{ \sum_{k\in K} \left[\bar{c}_a^k x_a^k + \mu (x_a^k)^2/2 \right] : \sum_{k\in K} x_a^k \le u_a, 11 \right\} \tag{16}$$

being $\bar{c}_a^k = c_a^k + \lambda_{a_+}^k - \lambda_{a_-}^k - \mu \bar{x}_a^k$. Now the crucial part is the *quadratic* KP (16), which is still easy to solve. Again, estimating the constant R_2, i.e., maximising the convex (13) over the feasible region, is not so. However, by the same token we maximise a linear upper approximation by solving the continuous KP

$$\bar{g}^a(\lambda) = \max \left\{ \sum_{k\in K} \left(u_a^k/2 - \bar{x}_a^k \right) : \sum_{k\in K} x_a^k \le u_a, (11) \right\}$$

and using $\bar{g}^a(\lambda)$ similarly to $g^a(\lambda)$ to provide an upper estimate to R_2.

4 Numerical experiments

The FG method has been developed in C++, compiled with GNU `g++ 4.4.5` (with `-O3` optimization option) and ran on an Opteron 6174 processor (2.2 GHz) with 32 GB of RAM, under a i686 GNU/Linux operating system. The solvers for *quadratic* MCF (14) and KP (16) are available thanks to the `MCFClass` and `CQKnPClass` projects, respectively, available at http://www.di.unipi.it/optimize/Software/MCF.html, http://www.di.unipi.it/optimize/Software/CQKnP.html.

The numerical experiments have been performed on 80 randomly generated instances already used in several papers [8,9], and available at http://www.di.unipi.it/optimize/Data/MMCF.html#Canad.

The purpose of the testing is to compare the static rule (5) proposed in [13] with the dynamic rule (6) making use of f_{lb}. To compare different algorithms we report convergence charts plotting the obtained relative gap, $(f_k^{best} - f_*)/|f_*|$, against both iteration and time. As in [9], the time charts for different instances become almost indistinguishable when the horizontal axis represents the *normalized time*, i.e., the running time divided by the product $|A| \cdot |K|$. This is illustrated in the right part of Fig. 1 (in the left one, the horizontal axis represents iterations) where convergence charts are separately reported, averaged on small instances ($|A| \leq 300$), medium ones ($300 < |A| \leq 600$) and large ones ($|A| > 600$): the individual lines are barely distinguishable among them and with the total average. The normalized time plots are a bit more apart from each other, which is reasonable because (14) and (16) are "complex" subproblems that cannot be expected to scale linearly with size, but still the difference is not large. As this consistently happens in all cases, in the following, we only report the global average.

We start by discussing the KR. In Figs. 2 and 3 we report the (average) convergence plots for the static rule (5) and the dynamic rule (6) when the lower bound is "accurate", i.e., $f_{lb} = f_*$ and, respectively, $\epsilon_r = 1e^{-4}$ and $\epsilon_r = 1e^{-6}$. As before, on the left side we plot the gap against the number of iterations, and on the right side against normalised time. To better put the results in perspective we also report results for two highly tuned version of the subgradient algorithm applied to the standard (non-smoothed) Lagrangian dual, using *volume* deflection and, respectively, *FumeroTV* (SVF) and *colorTV* (SVC) stepsize rules, with the best algorithmic parameters found in [9].

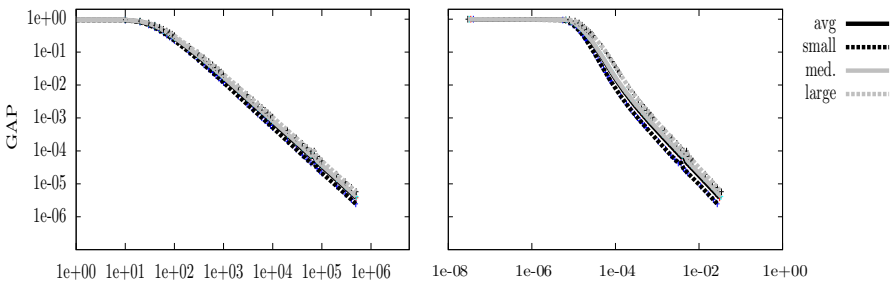

Fig. 1 Partly disaggregated results for dynamic μ with $f_{lb} = f_*$

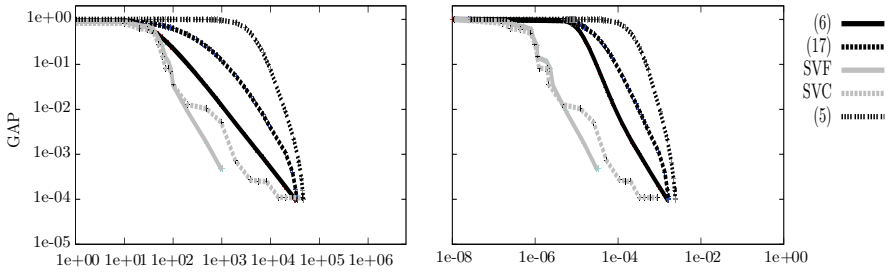

Fig. 2 Results for the KR with $f_{lb} = f_*$ and $\epsilon_r = 1e^{-4}$

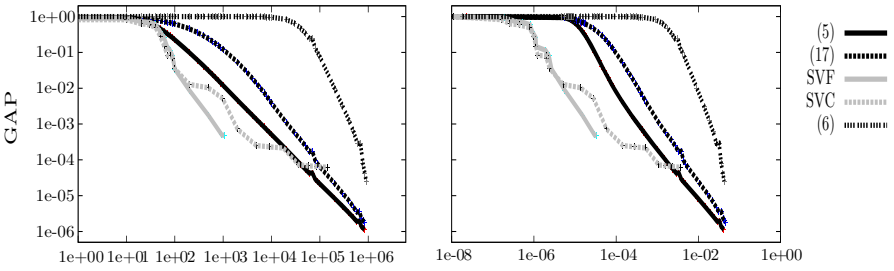

Fig. 3 Results for the KR with $f_{lb} = f_*$ and $\epsilon_r = 1e^{-6}$

Because we know a (tight) bound on the optimal value, we can stop all variants as soon as an accurate enough solution has been found, i.e., $f_k^{best} - f_* \leq \epsilon_r |f_*|$.

The figures clearly show that the dynamic rule (6) significantly outperforms the static one (5). In particular, the convergence plots show a first "flat" leg where progress is slow; comparing Figs. 2 and 3 (purposely plotted in identical horizontal scale) shows that the flat leg for (5) with $\epsilon_r = 1e^{-6}$ is much longer than with $\epsilon_r = 1e^{-4}$. This is somewhat unsettling, in that the final desired accuracy should not, in principle, influence the convergence speed at the beginning; yet it does for the static rule. The dynamic one attains, after a shorter flat leg, a remarkably linear convergence rate which is (correctly) not influenced by the value of ϵ_r. The FG with dynamic rule is roughly competitive with the subgradient variants (which also exploit knowledge of f_* for computing the stepsize) for $\epsilon_r = 1e^{-4}$, despite having to solve a more complex Lagrangian problem. The convergence profile of subgradient methods is considerably more erratic than that of the FG. Furthermore, they are basically incapable of attaining accuracy greater than $\epsilon_r = 1e^{-4}$ (and not even that for SVF), whereas the FG has no issues to get to $\epsilon_r = 1e^{-6}$, and likely beyond.

However, the picture is different when $f_{lb} \ll f_*$, as Figs. 4 and 5 show. There we use the significantly worse estimate for $f_{lb} = f_* - 0.1|f_*|$ (denoted as "10% f_*" for short). The result is that the dynamic rule "flattens out" far from the required accuracy, basically ceasing to converge. This is due to the fact that in (6) μ_k only becomes small if f_k^{best} approaches f_{lb}, which cannot happen because $f_{lb} \ll f_*$. Hence, μ is never set to the value required for attaining an accurate solution, and the FG basically stalls. Note that in the figures we plot two different versions of the static rule (5): (5') uses

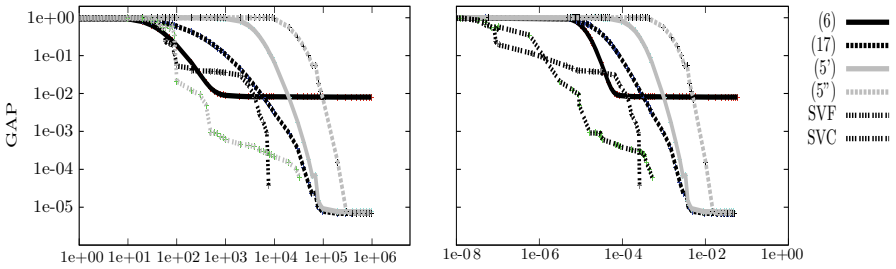

Fig. 4 Results for the KR with $f_{lb} = 10\% f_*$ and $\epsilon_r = 1e^{-4}$

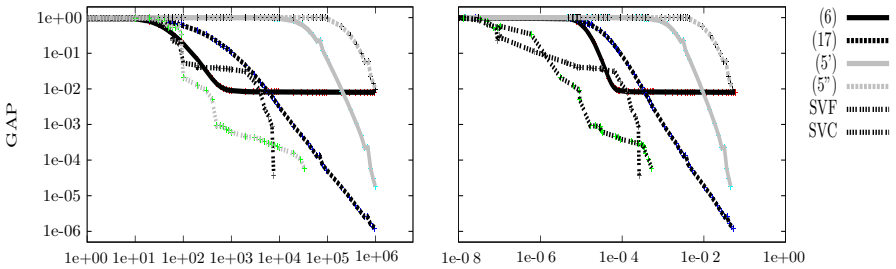

Fig. 5 Results for the KR with $f_{lb} = 10\% f_*$ and $\epsilon_r = 1e^{-6}$

$f_{ref} = f_{lb}$, while (5") uses $f_{ref} = f_k^{best}$. The first option turns out to be preferable, but both versions show the "flat leg" that grows longer as the required accuracy increases. A possible approach to remedy this drawback of the dynamic rule is to observe that, when $f_{lb} - f_*$, the convergence rate becomes very nearly linear on a doubly-logarithmic scale from a certain iteration $\hat{\imath}$ onwards. In other words, experimentally

$$\left[\log\left((f(\lambda_i) - f_*)/f_*\right) - \log\left((f(\lambda_{\hat{\imath}}) - f_*)/f_*\right)\right]/[\log(i) - \log(\hat{\imath})] = -\alpha$$

holds with quite good accuracy for all i larger than a properly chosen $\hat{\imath}$. This immediately suggests the empiric formula

$$\mu_k = \max\{\min\{(f_{\hat{\imath}} - f_{lb})(\hat{\imath}/k)^{\alpha}\,,\ (f_k^{best} - f_{lb})\}\,,\ \epsilon_r |f_{lb}|\}/(2R_2) \qquad (17)$$

for dynamically adjusting μ when f_{lb} might not be an accurate estimate of f_*. The parameters $\alpha = 1.2$ and $\hat{\imath} = 10$ are easily derived from the (average) convergence plot for $f_{lb} = f^*$, and used uniformly for all instances (being the convergence plots almost identical). Figures 2 and 3 show that the new dynamic strategy (17), albeit not as efficient as (6) with the accurate estimate of f_*, is still consistently superior to the static strategy (5). Furthermore, it is resilient to rather inaccurate estimates of f_*; indeed, it is by far the preferable option in Figs. 4 and 5.

The results for the FR are analogous, with a few differences. First of all, the quadratic MCF solvers had numerical issues with small values of μ, preventing us to reliably obtain runs for $\epsilon_r = 1e^{-6}$, which is why we only report results for $\epsilon_r = 1e^{-4}$. Second, according to [9], the best subgradient variant for this problem rather uses a

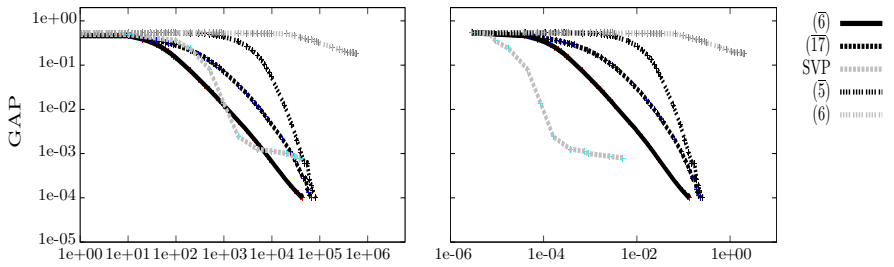

Fig. 6 Results for the FR with $f_{lb} = f_*$ and $\epsilon_r = 1e^{-4}$

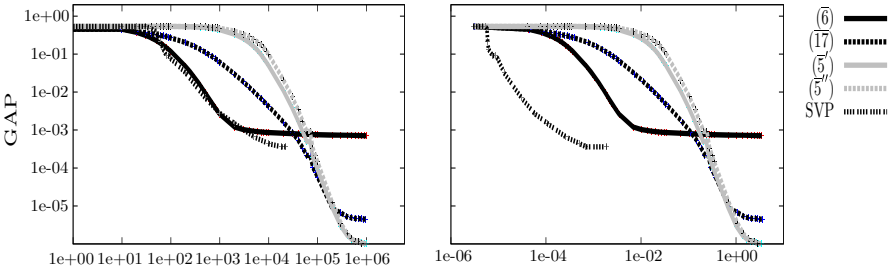

Fig. 7 Results for the FR with $f_{lb} = 10\% f_*$ and $\epsilon_r = 1e^{-4}$

Polyak stepsize rule (SVP). Finally, using the actual value of $\|B\|$ corresponding to (14)–(15) actually led to a surprisingly slow convergence. We (basically, by chance) discovered that using $\|B\| = 1$ instead recovered a much faster convergence. While this suggests that the FG may benefit from some tuning, exploring this issue is out of the scope of the present paper. Therefore, in Figs. 6 and 7, we mainly report the results of the three rules when using $\|B\| = 1$, denoted by $(\overline{5})$, $(\overline{6})$ and $(\overline{17})$, while only plotting in Fig. 6, the results of the original rule (6) to show how much worse the performances are (those of the other rules are similarly degraded).

All in all, the results closely mirror those of the KR. The subgradient method is considerably faster than FG, more so than in the KR, which is not surprising because quadratic MCFs now have to be solved; however, it struggles to reach $\epsilon_r = 1e^{-4}$ accuracy. The dynamic rule (6) is preferable when $f_{lb} = f_*$, but it stalls far from the required accuracy when the lower bound is not accurate, in which case the dynamic rule (6) is preferable. In general, the static rule (5), in both variants, is less effective than the dynamic ones. The exception is at the end of the convergence plot in Fig. 7; however, this corresponds to the case where the desired accuracy has already been attained, but the FG is not capable of stopping (quickly) because the lower bound is not accurate enough. Only in that final phase the static strategy outperforms the dynamic one.

5 Conclusion

We have devised a simple rule for dynamically adjusting the crucial smoothness parameter μ in the fast gradient approach. The rule exploits information about the optimal

value of the problem to significantly improve the convergence properties of the method, at least in practice on our test instances. The rule is very effective when the estimate is tight, but it can also be adapted to work when the estimate is loose. This requires tuning two parameters, which in our experience seems to be easy. The proposed modification is therefore interesting for all the applications where bounds on the optimal value are readily available, as it happens, e.g., in integer optimization. Besides possibly proving useful for various applications that can benefit from FG approaches, we hope that our result stimulates research into finding ways for exploiting information about the optimal function value in the related, although different, primal-dual subgradient methods (PDSM) [12] that do not require modifying the function computation to work. The inability to exploit this information has been identified as a potential weakness in PDSM [9], which limits the applicability of this otherwise interesting—both for its performances and for being almost parameter-free—class of subgradient algorithms. Our results on FG seem to indicate that this line of research could bear interesting fruits.

References

1. Ahookhosh, M., Neumaier, A.: Optimal subgradient algorithms for large-scale convex optimization in simple domains. Numer. Algorithms (2017). doi:10.1007/s11075-017-0297-x
2. Beck, A., Teboulle, M.: Smoothing and first order methods: a unified framework. SIAM J. Optim. **22**(2), 557–580 (2012)
3. Bot, R., Hendrich, C.: A variable smoothing algorithm for solving convex optimization problems. TOP **23**(1), 124–150 (2015)
4. Chambolle, A., Pock, T.: A first-order primal-dual algorithm for convex problems with applications to imaging. J. Math. Imaging Vis. **40**(1), 120–145 (2011)
5. Chouman, M., Crainic, T., Gendron, B.: Commodity representations and cut-set-based inequalities for multicommodity capacitated fixed-charge network design. Transp. Sci. **51**(2), 650–667 (2017)
6. d'Antonio, G., Frangioni, A.: Convergence analysis of deflected conditional approximate subgradient methods. SIAM J. Optim. **20**(1), 357–386 (2009)
7. Frangioni, A.: Generalized bundle methods. SIAM J. Optim. **13**(1), 117–156 (2002)
8. Frangioni, A., Gorgone, E.: Generalized bundle methods for sum-functions with "easy" components: applications to multicommodity network design. Math. Program. **145**(1), 133–161 (2014)
9. Frangioni, A., Gorgone, E., Gendron, B.: On the computational efficiency of subgradient methods: a case study with lagrangian bounds. Math. Program. Comput. (2017). doi:10.1007/s12532-017-0120-7
10. Hiriart-Urruty, J.B., Lemaréchal, C.: Convex Analysis and Minimization Algorithms II—Advanced Theory and Bundle Methods. Grundlehren der mathematischen Wissenschaften, vol. 306. Springer, New York (1993)
11. Lan, G., Zhou, Y.: Approximation accuracy, gradient methods, and error bound for structured convex optimization. Technical report, University of Florida (2014)
12. Nesterov, Y.: Primal-dual subgradient methods for convex optimization. SIAM J. Optim. **12**, 109–138 (2001)
13. Nesterov, Y.: Smooth minimization of non-smooth functions. Math. Program. **103**, 127–152 (2005)
14. Shor, N.: Minimization Methods for Nondifferentiable Functions. Springer, Berlin (1985)