Conditional Gradient Algorithms for Norm-Regularized Smooth Convex Optimization

Zaid Harchaoui, Anatoli B. Juditsky, Arkadii S. Nemirovski

Conditional Gradient Algorithms for Norm-Regularized Smooth Convex Optimization

Zaid Harchaoui * Anatoli Juditsky† Arkadi Nemirovski‡

Abstract

Motivated by some applications in signal processing and machine learning, we consider two convex optimization problems where, given a cone K, a norm $\|\cdot\|$ and a smooth convex function f, we want either 1) to minimize the norm over the intersection of the cone and a level set of f, or 2) to minimize over the cone the sum of f and a multiple of the norm. We focus on the case where (a) the dimension of the problem is too large to allow for interior point algorithms, (b) $\|\cdot\|$ is "too complicated" to allow for computationally cheap Bregman projections required in the first-order proximal gradient algorithms. On the other hand, we assume that it is relatively easy to minimize linear forms over the intersection of K and the unit $\|\cdot\|$-ball. Motivating examples are given by the nuclear norm with K being the entire space of matrices, or the positive semidefinite cone in the space of symmetric matrices, and the Total Variation norm on the space of 2D images. We discuss versions of the Conditional Gradient algorithm capable to handle our problems of interest, provide the related theoretical efficiency estimates and outline some applications.

1 Introduction

We consider two *norm-regularized* convex optimization problems as follows:

$$\text{[norm minimization]} \qquad \min_{x\in K} \|x\|, \ \text{ subject to } \ f(x) \le \delta, \tag{1}$$

$$\text{[penalized minimization]} \qquad \min_{x\in K} f(x) + \kappa\|x\| \tag{2}$$

where f is a convex function with Lipschitz continuous gradient, K is a closed convex cone in a Euclidean space E, $\|\cdot\|$ is some norm, δ and κ are positive parameters. Problems such as such as (1) and (2) are of definite interest for signal processing and machine learning. In these applications, $f(x)$ quantifies the discrepancy between the observed noisy output of some parametric model and the output of the model with candidate vector x of parameters. Most notably, f is the quadratic penalty: $f(x) = \frac{1}{2}\|\mathcal{A}x - y\|_2^2$, where $\mathcal{A}x$ is the "true" output of the linear regression model $x \mapsto \mathcal{A}x$, and $y = \mathcal{A}x_* + \xi$, where x_* is the vector of true parameters, ξ is the observation error, and δ is an a priori upper bound on $\frac{1}{2}\|\xi\|_2^2$. The cone K sums up a priori information on the parameter vectors (e.g., $K = E$ – no a priori information at all, or $E = \mathbf{R}^p$, $K = \mathbf{R}^p_+$, or $E = \mathbf{S}^p$, the space of

*LJK, INRIA Rhône-Alpes, 655 Avenue de l'Europe, Montbonnot, 38334 Saint-Ismier France zaid.harchaoui@inria.fr

†LJK, Université J. Fourier, B.P. 53, 38041 Grenoble Cedex 9, France, anatoli.juditsky@imag.fr

‡Georgia Institute of Technology, Atlanta, Georgia 30332, USA, nemirovs@isye.gatech.edu

Research of the third author was supported by the ONR grant N000140811104 and NSF grants DMS 0914785, CMMI 1232623

symmetric $p \times p$ matrices, and $K = \mathbf{S}^p_+$, the cone of positive semidefinite matrices, as is the case of covariance matrices recovery). Finally, $\|\cdot\|$ is a regularizing norm "promoting" a desired property of the recovery, e.g., the sparsity-promoting norm ℓ_1 on $E = \mathbf{R}^n$, or the low rank promoting nuclear norm on $E = \mathbf{R}^{p\times q}$, or the Total Variation (TV) norm, as in image reconstruction.

In the large-scale case, first-order algorithms of proximal-gradient type are popular to tackle such problems, see [30] for a recent overview. Among them, the celebrated Nesterov optimal gradient methods for smooth and composite minimization [22, 23, 24], and their stochastic approximation counterparts [18], are now state-of-the-art in compressive sensing and machine learning. These algorithms enjoy the best known so far theoretical estimates (and in some cases, these estimates are the best possible for the first-order algorithms). For instance, Nesterov's algorithm for penalized minimization [23, 24] solves (2) to accuracy ϵ in $O(D_0\sqrt{L/\epsilon})$ iterations, where L is the properly defined Lipschitz constant of the gradient of f, and D_0 is the initial distance to the optimal set, measured in the norm $\|\cdot\|$. However, applicability and efficiency of proximal-gradient algorithms in the large-scale case require from the problem to possess "favorable geometry" (for details, see [24, Section A.6]). To be more specific, consider proximal-gradient algorithm for convex minimization problems of the form

$$\min_x \{f(x) : \|x\| \le 1, x \in K\}. \tag{3}$$

The comments to follow, with slight modifications, are applicable to problems such as (1) and (2) as well. In this case, a proximal-gradient algorithm operates with a "distance generating function" (d.g.f.) defined on the domain of the problem and 1-strongly convex w.r.t. the norm $\|\cdot\|$. Each step of the algorithm requires minimizing the sum of the d.g.f. and a linear form. The efficiency estimate of the algorithm depends on the variation of the d.g.f. on the domain and on regularity of f w.r.t. $\|\cdot\|$ [1]. As a result, in order for a proximal-gradient algorithm to be practical in the large scale case, two "favorable geometry" conditions should be met: (a) the outlined sub-problems should be easy to solve, and (b) the variation of the d.g.f. on the domain of the problem should grow slowly (if at all) with problem's dimension. Both these conditions indeed are met in many applications; see, e.g., [2, 17] for examples. This explains the recent popularity of this family of algorithms.

However, sometimes conditions (a) and/or (b) are violated, and application of proximal algorithms becomes questionable. For example, for the case of $K = E$, (b) is violated for the usual $\|\cdot\|_\infty$-norm on $\mathbf{R}^p$ or, more generally, for $\|\cdot\|_{2,1}$ norm on the space of $p \times q$ matrices given by

$$\|x\|_{2,1} = \max_{1\le j\le p} \|\mathrm{Row}_j(x)\|_2,$$

where $\mathrm{Row}_j^T(x)$ denotes the j-th row of x. Here the variation of (any) d.g.f. on problem's domain is at least p. As a result, in the case in question the theoretical iteration complexity of a proximal algorithm grows rapidly with the dimension p. Furthermore, for some high-dimensional problems which do satisfy (b), solving the sub-problem can be computationally challenging. Examples of such problems include nuclear-norm-based matrix completion, Total Variation-based image reconstruction, and multi-task learning with a large number of tasks and features. This corresponds to $\|\cdot\|$ in (1) or (2) being the nuclear norm [10] or the TV-norm.

[1] i.e., the Lipschitz constant of f w.r.t. $\|\cdot\|$ in the nonsmooth case, or the Lipschitz constant of the gradient mapping $x \mapsto f'(x)$ w.r.t. the norm $\|\cdot\|$ on the argument and the conjugate of this norm on the image spaces in the smooth case.

These limitations recently motivated alternative approaches, which do not rely upon favorable geometry of the problem domain and/or do not require to solve hard sub-problems at each iteration, and triggered a renewed interest in the Conditional Gradient (CndG) algorithm. This algorithm, also known as the Frank-Wolfe algorithm, which is historically the first method for smooth constrained convex optimization, originates from [8], and was extensively studied in the 70-s (see, e.g., [5, 7, 25] and references therein). CndG algorithms work by minimizing a linear form on the problem domain at each iteration; this auxiliary problem clearly is easier, and in many cases – significantly easier than the auxiliary problem arising in proximal-gradient algorithms. Conditional gradient algorithms for collaborative filtering were studied recently [15, 14], some variants and extensions were studied in [6, 29, 10]. Those works consider constrained formulations of machine learning or signal processing problems, i.e., minimizing the discrepancy $f(x)$ under a constraint on the norm of the solution, as in (3). On the other hand, CndG algorithms for other learning formulations, such as norm minimization (1) or penalized minimization (2) remain open issues. An exception is the work of [6, 10], where a Conditional Gradient algorithm for penalized minimization was studied, although the efficiency estimates obtained in that paper were suboptimal. In this paper, we present CndG-type algorithms aimed at solving norm minimization and penalized norm minimization problems and provide theoretical efficiency guarantees for these algorithms.

The main body of the paper is organized as follows. In Section 2, we present detailed setting of problems (1), (2) along with basic assumptions on the "computational environment" required by the CndG-based algorithms we are developing. These algorithms and their efficiency bounds are presented in Sections 3 (problem (1)) and 5 (problem (2). In Section 6 we outline some applications, and in Section 7 present preliminary numerical results. All proofs are relegated to the appendix.

2 Problem statement

Throughout the paper, we shall assume that $K \subset E$ is a closed convex cone in Euclidean space E; we loose nothing by assuming that K linearly spans E. We assume, further, that $\|\cdot\|$ is a norm on E, and $f : K \to \mathbf{R}$ is a convex function with Lipschitz continuous gradient, so that

$$\|f'(x) - f'(y)\|_* \leq L_f \|x - y\| \ \forall x, y \in K,$$

where $\|\cdot\|_*$ denotes the norm dual to $\|\cdot\|$, whence

$$\forall x, y \in K : f(y) \leq f(x) + \langle f'(x), y - x\rangle + \frac{L_f}{2}\|y - x\|^2. \tag{4}$$

We consider two kinds of problems, detailed below.

Norm-minimization. Such problems correspond to

$$\rho_* = \min_x \{\|x\| : \ x \in K, \ f(x) \leq 0\}. \tag{5}$$

To tackle (5), we consider the following parametric family of problems

$$\mathrm{Opt}(\rho) = \min\{f(x) : \ \|x\| \leq \rho, \ x \in K\}. \tag{6}$$

Note that whenever (5) is feasible, which we assume from now on, we have

$$\rho_* = \min\{\rho \geq 0 : \ \mathrm{Opt}(\rho) \leq 0\}, \tag{7}$$

and both problems (5), (7) can be solved.

Given a tolerance $\epsilon > 0$, we want to find an ϵ-solution to the problem, that is, a pair ρ_ϵ, $x_\epsilon \in K$ such that

$$\rho_\epsilon \le \rho_* \text{ and } x_\epsilon \in X_{\rho_\epsilon} \text{ such that } f(x_\epsilon) \le \epsilon, \tag{8}$$

where $X_\rho := \{x \in E : \ x \in K, \ \|x\| \le \rho\}$. Getting back to the problem of interest (5), x_ϵ is then "super-optimal" and ϵ-feasible:

$$\|x_\epsilon\| \le \rho_\epsilon \le \rho_*, \quad f(x_\epsilon) \le \epsilon.$$

Penalized norm minimization. These problems write as

$$\text{Opt} = \min_x \{f(x) + \kappa\|x\| : \ x \in K\} \ . \tag{9}$$

A equivalent formulation is

$$\text{Opt} = \min_{x,r} \{F([x;r]) = \kappa r + f(x) : x \in K, \|x\| \le r\} \, . \tag{10}$$

We shall refer to (10) as the problem of *composite optimization* (CO). Given a tolerance $\epsilon > 0$, our goal is to find an ϵ-solution to (10), defined as a feasible solution (x_ϵ, r_ϵ) to the problem satisfying $F([x_\epsilon; r_\epsilon]) - \text{Opt} \le \epsilon$. Note that in this case x_ϵ is an ϵ-solution, in the similar sense, to (9).

Special case. In many applications where Problem (5) arise, (9) the function f enjoys a special structure:

$$f(x) = \phi(\mathcal{A}x - b),$$

where $x \mapsto \mathcal{A}x - b$ is an affine mapping from E to $\mathbf{R}^m$, and $\phi(\cdot) : \mathbf{R}^m \to \mathbf{R}$ is a convex function with Lipschitz continuous gradient; we shall refer to this situation as to *special case*. In such case, the quantity L_f can be bounded as follows. Let $\pi(\cdot)$ be some norm on $\mathbf{R}^m$, $\pi_*(\cdot)$ be the conjugate norm, and $\|\mathcal{A}\|_{\|\cdot\|,\pi}$ be the norm of the linear mapping $x \mapsto \mathcal{A}x$ induced by the norms $\|\cdot\|$, $\pi(\cdot)$ on the argument and the image spaces:

$$\|\mathcal{A}\|_{\|\cdot\|,\pi(\cdot)} = \max_{x \in E} \{\pi(\mathcal{A}x) : \|x\| \le 1\}.$$

Let also $L_{\pi(\cdot)}[\phi]$ be the Lipschitz constant of the gradient of ϕ induced by the norm $\pi(\cdot)$, so that

$$\pi_*(\phi'(y) - \phi'(y')) \le L_{\pi(\cdot)}[\phi]\pi(y - y') \ \forall y, y' \in F.$$

Then, one can take as L_f the quantity

$$L_f = L_{\pi(\cdot)}[\phi]\|\mathcal{A}\|^2_{\|\cdot\|,\pi(\cdot)}. \tag{11}$$

Example 1: quadratic fit. In many applications, we are interested in $\|\cdot\|_2$-discrepancy between $\mathcal{A}x$ and b; the related choice of $\phi(\cdot)$ is $\phi(y) = \frac{1}{2}y^T y$. Specifying $\pi(\cdot)$ as $\|\cdot\|_2$, we get $L_{\|\cdot\|_2}[\phi] = 1$.
Example 2: smoothed ℓ_∞ fit. When interested in $\|\cdot\|_\infty$ discrepancy between $\mathcal{A}x$ and b, we can use as ϕ the function $\phi(y) = \frac{1}{2}\|y\|^2_\beta$, where $\beta \in [2, \infty)$. Taking $\pi(\cdot)$ as $\|\cdot\|_\infty$, we get

$$L_{\|\cdot\|_\infty}[\phi] \le (\beta - 1)m^{2/\beta}.$$

Note that

$$\frac{1}{2}\|y\|_\infty^2 \le \phi(y) \le \frac{m^{2/\beta}}{2}\|y\|_\infty^2,$$

so that for $\beta = O(1)\ln(m)$ and m large enough (specifically, such that $\beta \ge 2$), $\phi(y)$ is within absolute constant factor of $\frac{1}{2}\|y\|_\infty^2$. The latter situation can be interpreted as ϕ behaving as $\frac{1}{2}\|\cdot\|_\infty^2$). At the same time, with $\beta = O(1)\ln(m)$, $L_{\|\cdot\|_\infty}[\phi] \le O(1)\ln(m)$ grows with m logarithmically.

Another widely used choice of $\phi(\cdot)$ for this type of discrepancy is "logistic" function

$$\phi(y) = \frac{1}{\beta}\ln\left(\sum_{i=1}^m \left[e^{\beta y_i} + e^{-\beta y_i}\right]\right).$$

For $\pi(\cdot) = \|\cdot\|_\infty$ we easily compute $L_{\|\cdot\|_\infty}[\phi] \le \beta$ and $\|y\|_\infty \le \phi(y) \le \|y\|_\infty + \ln(2n)/\beta$.

Note that in some applications we are interested in "one-sided" discrepancies quantifying the magnitude of the vector $[\mathcal{A}x - b]_+ = [\max[0, (\mathcal{A}x - b)_1]; ...; \max[0, (\mathcal{A}x - b)_m]]$ rather than the the magnitude of the vector $\mathcal{A}x - b$ itself. Here, instead of using $\phi(y) = \frac{1}{2}\|y\|_\beta^2$ in the context of examples 1 and 2, one can use the functions $\phi_+(y) = \phi([y]_+)$. In this case the bounds on $L_{\pi(\cdot)}[\phi_+]$ are exactly the same as the above bounds on $L_{\pi(\cdot)}[\phi]$. The obvious substitute for the two-sided logistic function is its "one-sided version:" $\phi_+(y) = \frac{1}{\beta}\ln\left(\sum_{i=1}^m \left[e^{\beta y_i} + 1\right]\right)$ which obeys the same bound for $L_{\pi(\cdot)}[\phi_+]$ as its two-sided analogue.

First-order and Linear Optimization oracles. We assume that f is represented by a *first-order oracle* – a routine which, given on input a point $x \in K$, returns the value $f(x)$ and the gradient $f'(x)$ of f at x. As about K and $\|\cdot\|$, we assume that they are given by a *Linear Optimization (LO) oracle* which, given on input a linear form $\langle\eta, \cdot\rangle$ on E, returns a minimizer $x[\eta]$ of this linear form on the set $\{x \in K : \|x\| \le 1\}$. We assume w.l.o.g. that for every η, $x[\eta]$ is either zero, or is a vector of the $\|\cdot\|$-norm equal to 1. To ensure this property, it suffices to compute $\langle\eta, x[\eta]\rangle$ for $x[\eta]$ given by the oracle; if this inner product is 0, we can reset $x[\eta] = 0$, otherwise $\|x[\eta]\|$ is automatically equal to 1.

Note that an LO oracle for K and $\|\cdot\|$ allows to find a minimizer of a linear form of $z = [x; r] \in E^+ := E \times \mathbf{R}$ on a set of the form $K^+[\rho] = \{[x; r] \in E^+ : x \in K, \|x\| \le r \le \rho\}$ due to the following observation:

Lemma 1. *Let $\rho \ge 0$ and $\eta^+ = [\eta; \sigma] \in E^+$. Consider the linear form $\ell(z) = \langle\eta^+, z\rangle$ of $z = [x; r] \in E^+$, and let*

$$z^+ = \begin{cases} \rho[x[\eta]; 1] & , \langle\eta^+, [x[\eta]; 1]\rangle \le 0, \\ 0 & , \textit{otherwise} \end{cases}.$$

Then z^+ is a minimizer of $\ell(z)$ over $z \in K^+[\rho]$, When $\sigma = 0$, one has $z^+ = \rho[x[\eta]; 1]$.

Indeed, let $z_* = [x^*; r_*]$ be a minimizer of $\ell(\cdot)$ over $K^+[\rho]$. Since $\|x^*\| \le r_*$ due to $[x^*; r_*] \in K^+[\rho]$, we have $z^* := r^*[x[\eta]; 1] \in K^+[\rho]$ due to $\|x[\eta]\| \le 1$, and $\ell * (z^*) \le \ell(z_*)$ due to the definition of $x[\eta]$. We conclude that any minimizer of $\ell(\cdot)$ over the segment $\{s[x[\eta]; 1] : 0 \le s \le \rho\}$ is also a minimizer of $\ell(\cdot)$ over $K^+[\rho]$. It remains to note that the vector indicated in Lemma clearly is a minimizer of $\ell(\cdot)$ on the above segment. □

3 Conditional Gradient algorithm

In this section, we present an overview of the properties of the standard Conditional Gradient algorithm, and highlight some memory-based extensions. These properties are not new. However, since they are key for the design of our proposed algorithms in the next sections, we present them for further reference.

3.1 Conditional gradient algorithm

Let E be a Euclidean space and X be a closed and bounded convex set in E which linearly spans E. Assume that X is given by a LO oracle – a routine which, given on input $\eta \in E$, returns an optimal solution $x_X[\eta]$ to the optimization problem

$$\min_{x \in X} \langle \eta, x \rangle$$

(cf. Section 2). Let f be a convex differentiable function on X with Lipschitz continuous gradient $f'(x)$, so that

$$\forall x, y \in X : f(y) \leq f(x) + \langle f'(x), y - x \rangle + \tfrac{1}{2} L \|y - x\|_X^2, \tag{12}$$

where $\|\cdot\|_X$ is the norm on E with the unit ball $X - X$. We intend to solve the problem

$$f_* = \min_{x \in X} f(x). \tag{13}$$

A *generic CndG algorithm* is a recurrence which builds iterates $x_t \in X$, $t = 1, 2, ...$, in such a way that

$$f(x_{t+1}) \leq f(\widetilde{x}_{t+1}), \tag{14}$$

where

$$\widetilde{x}_{t+1} \quad = \quad x_t + \gamma_t [x_t^+ - x_t], \quad \text{where } x_t^+ = x_X[f'(x_t)] \text{ and } \gamma_t = \tfrac{2}{t+1}. \tag{15}$$

Basic implementations of a generic CndG algorithm are given by

$$\begin{array}{lrcl} (a) & x_{t+1} & = & x_t + \gamma_t [x_t^+ - x_t], \ \ \gamma_t = \frac{2}{t+1}, \\ (b) & x_{t+1} & \in & \mathrm{Argmin}_{x \in D_t} f(x), \ \ D_t = [x_t, x_t^+]; \end{array} \tag{16}$$

in the sequel, we refer to them as CndGa and CndGb, respectively. As a byproduct of running generic CndG, after t steps we have at our disposal the quantities

$$f_{*,k} = \min_{x \in X} \big[f(x_k) + \langle f'(x_k), x - x_k \rangle \big] = f(x_k) - \langle f'(x_k), x_k - x_X[f'(x_k)] \rangle, \ 1 \leq k \leq t, \tag{17}$$

which, by convexity of f, are lower bounds on f_*. Consequently, at the end of step t we have at our disposal a lower bound

$$f_*^t := \max_{1 \leq k \leq t} f_{*,k} \leq f_*, \ t = 1, 2, ... \tag{18}$$

on f_*.

Finally, we define the approximate solution $\bar{x}_t$ found in course of $t = 1, 2, ...$ steps as the best – with the smallest value of f – of the points $x_1, ..., x_t$. Note that $\bar{x}_t \in X$.

The following statement summarizes the well known properties of CndG (to make the presentation self-contained, we provide in Appendix the proof).

Theorem 1. *For a generic CndG algorithm, in particular, for both CndGa, CndGb, we have*

$$f(\bar{x}_t) - f_* \le f(x_t) - f_* \le \frac{2L}{t+1}, \quad t \ge 2; \tag{19}$$

and

$$f(\bar{x}_t) - f_*^t \le \frac{4.5L}{t-2}, \quad t \ge 5. \tag{20}$$

Some remarks regarding the conditional algorithm are in order.

Certifying quality of approximate solutions. An attractive property of CndG is the presence of online lower bound f_*^t on f_* which certifies the theoretical rate of convergence of the algorithm, see (20). This accuracy certificate, first established in [14], also provides a valuable stopping criterion when running the algorithm in practice.

CndG algorithm with memory. When computing the next search point x_{t+1} the simplest CndG algorithm CndGa only uses the latest answer $x_t^+ = x_X[f'(x_t)]$ of the LO oracle. Meanwhile, algorithm CndGb can be modified to make use of information supplied by previous oracle calls; we refer to this modification as CndG with memory (CndGM).[2]. Assume that we have already carried out $t-1$ steps of the algorithm and have at our disposal current iterate $x_t \in X$ (with x_1 selected as an arbitrary point of X) along with previous iterates x_τ, $\tau < t$ and the vectors $f'(x_\tau)$, $x_\tau^+ = x_X[f'(x_\tau)]$. At the step, we compute $f'(x_t)$ and $x_t^+ = x_X[f'(x_t)]$. Thus, at this point in time we have at our disposal $2t$ points x_τ, x_τ^+, $1 \le \tau \le t$, which belong to X. Let X_t be subset of these points, with the only restriction that the points x_t, x_t^+ are selected, and let us define the next iterate x_{t+1} as

$$x_{t+1} \in \operatorname*{Argmin}_{x \in \mathrm{Conv}(X_t)} f(x), \tag{21}$$

that is,

$$x_{t+1} = \sum_{x \in X_t} \lambda_x^t x, \ \lambda^t \in \operatorname*{Argmin}_{\lambda^t = \{\lambda_x\}_{x \in X_t}} \left\{ f\left(\sum\nolimits_{x \in X_t} \lambda_x x\right) : \ \lambda \ge 0, \ \sum_{x \in X_t} \lambda_x = 1 \right\}. \tag{22}$$

Clearly, it is again a generic CndG algorithm, so that conclusions in Theorem 1 are fully applicable to CndGM. Note that CndGb per se is nothing but CndGM with $X_t = \{x_t, x_t^+\}$ and $M = 2$ for all t.

CndGM: implementation issues. Assume that the cardinalities of the sets X_t in CndGM are bounded by some $M \ge 2$. In this case, implementation of the method requires solving at every step an auxiliary problem (22) of minimizing over the standard simplex of dimension $\le M-1$ a smooth convex function given by a first-order oracle induced by the first-oracle for f. When M is a once for ever fixed small integer, the arithmetic cost of solving this problem within machine accuracy by, say, the Ellipsoid algorithm is dominated by the arithmetic cost of just $O(1)$ calls to the first-order oracle for f. Thus, CndGM with small M can be considered as implementable[3].

[2]Note that in the context of "classical" Frank-Wolfe algorithm – minimization of a smooth function over a polyhedral set – such modification is referred to as *Restricted Simplicial Decomposition* [13, 12, 32]

[3]Assuming possibility to solve (22) exactly, while being idealization, is basically as "tolerable" as the standard in continuous optimization assumption that one can use exact real arithmetic or compute exactly eigenvalues/eigenvectors of symmetric matrices. The outlined "real life" considerations can be replaced with rigorous error analysis which shows that in order to maintain the efficiency estimates from Theorem 1, it suffices to solve t-th auxiliary problem within properly selected positive inaccuracy, and this can be achieved in $O(\ln(t))$ computations of f and f'.

Note that in the special case (Section 2), where $f(x) = \phi(Ax - b)$, assuming $\phi(\cdot)$ and $\phi'(\cdot)$ easy to compute, as is the case in most of the applications, the first-order oracle for the auxiliary problems arising in CndGM becomes cheap (cf. [34]). Indeed, in this case (22) reads

$$\min_{\lambda^t} \left\{ g_t(\lambda^t) := \phi\left(\sum\nolimits_{x\in X_t} \lambda_x^t Ax - b\right) : \begin{array}{l} \lambda^t = \{\lambda_x^t\}_{x\in X_t} \geq 0, \\ \sum_{x\in X_t} \lambda_x^t = 1 \end{array} \right\}.$$

It follows that all we need to get a computationally cheap access to the first-order information on $g_t(\lambda^t)$ *for all* values of λ^t is to have at our disposal the matrix-vector products Ax, $x \in X_t$. With our construction of X_t, the only two "new" elements in X_t (those which were not available at preceding iterations) are x_t and x_t^+, so that the only two new matrix-vector products we need to compute at iteration t are Ax_t (which usually is a byproduct of computing $f'(x_t)$) and Ax_t^+. Thus, we can say that the "computational overhead," as compared to computing $f'(x_t)$ and $x_t^+ = x_X[f'(x_t)]$, needed to get easy access to the first-order information on $g_t(\cdot)$ reduces to computing the single matrix-vector product Ax_t^+.

4 Conditional gradient algorithm for parametric optimization

In this section, we describe a multi-stage algorithm to solve the parametric optimization problem (6), (7), using the conditional algorithm to solve inner sub-problems. (6), (7). The idea, originating from [19] (see also [22, 16, 24]), is to use a Newton-type method for approximating from below the positive root ρ_* of $\mathrm{Opt}(\rho)$, with (inexact) first-order information on $\mathrm{Opt}(\cdot)$ yielded by approximate solving the optimization problems defining $\mathrm{Opt}(\cdot)$; the difference with the outlined references is that now we solve these problems with the CndG algorithm.

Our algorithm works stagewise. At the beginning of stage $s = 1, 2, ...$, we have at hand a lower bound ρ_s on ρ_*, with ρ_1 defined as follows:

> We compute $f(0)$, $f'(0)$ and $x[f'(0)]$. If $f(0) \leq \epsilon$ or $x[f'(0)] = 0$, we are done — the pair ($\rho = 0$, $x = 0$) is an ϵ-solution to (7) in the first case, and is an optimal solution to the problem in the second case (since in the latter case 0 is a minimizer of f on K, and (7) is feasible). Assume from now on that the above options do not take place ("nontrivial case"), and let
>
> $$d = -\langle f'(0), x[f'(0)]\rangle.$$
>
> Due to the origin of $x[\cdot]$, d is positive, and $f(x) \geq f(0) + \langle f'(0), x\rangle \geq f(0) - d\|x\|$ for all $x \in K$, which implies that $\rho_* \geq \rho_1 := \frac{f(0)}{d} > 0$.

At stage s we apply a generic CndG algorithm (e.g., CndGa,CndGb, or CndGM; in the sequel, we refer to the algorithm we use as to CndG) to the auxiliary problem

$$\mathrm{Opt}(\rho_s) = \min_x \{f(x) : x \in K[\rho_s]\}, \quad K[\rho] = \{x \in K : \|x\| \leq \rho\}, \tag{23}$$

Note that the LO oracle for K, $\|\cdot\|$ induces an LO oracle for $K[\rho]$; specifically, for every $\eta \in E$, the point $x_\rho[\eta] := \rho x[\eta]$ is a minimizer of the linear form $\langle \eta, x\rangle$ over $x \in K[\rho]$, see Lemma 1. $x_\rho[\cdot]$ is exactly the LO oracle utilized by CndG as applied to (23).

As explained above, after t steps of CndG as applied to (23), the iterates being $x_\tau \in K[\rho_s]$, $1 \le \tau \le t$ [4], we have at our disposal current approximate solution $\bar{x}_t \in \{x_1, ..., x_t\}$ such that $f(\bar{x}_t) = \min_{1\le\tau\le t} f(x_\tau)$ along with a lower bound f_*^t on $\mathrm{Opt}(\rho_s)$. Our policy is as follows.

1. When $f(\bar{x}_t) \le \epsilon$, we terminate the solution process and output $\bar{\rho} = \rho_s$ and $\bar{x} = \bar{x}_t$;
2. When the above option is not met and $f_*^t < \frac{3}{4} f(\bar{x}_t)$, we specify x_{t+1} according to the description of CndG and pass to step $t+1$ of stage s;
3. Finally, when neither one of the above options takes place, we terminate stage s and pass to stage $s+1$, specifying ρ_{s+1} as follows:
   We are in the situation $f(\bar{x}_t) > \epsilon$ and $f_*^t \ge \frac{3}{4} f(\bar{x}_t)$. Now, for $k \le t$ the quantities $f(x_k)$, $f'(x_k)$ and $x[f'(x_k)]$ define affine function of $\rho \ge 0$

$$\ell_k(\rho) = f(x_k) + \langle f'(x_k), x - \rho x[f'(x_k)]\rangle.$$

   By Lemma 1 we have for every $\rho \ge 0$

$$\ell_k(\rho) = \min_{x\in K[\rho]} \big[f(x_k) + \langle f'(x_k), x - x_k\rangle\big] \le \min_{x\in K[\rho]} f(x) = \mathrm{Opt}(\rho),$$

   where the inequality is due to the convexity of f. Thus, $\ell_k(\rho)$ is an affine in $\rho \ge 0$ lower bound on $\mathrm{Opt}(\rho)$, and we lose nothing by assuming that all these univariate affine functions are memorized when running CndG on (23). Note that by construction of the lower bound f_*^t (see (17), (18) and take into account that we are in the case of $X = K[\rho_s]$, $x_X[\eta] = \rho_s x[\eta]$) we have

$$f_*^t = \ell^t(\rho_s),\ \ell^t(\rho) = \max_{1\le k\le t} \ell_k(\rho).$$

   Note that $\ell^t(\rho)$ is a lower bound on $\mathrm{Opt}(\rho)$, so that $\ell^t(\rho) \le 0$ for $\rho \ge \rho_*$, while $\ell^t(\rho_s) = f_*^t$ is positive. It follows that

$$r^t := \min\big\{\rho : \ell^t(\rho) \le 0\big\}$$

   is well defined and satisfies $\rho_s < r^t \le \rho_*$. We compute r^t (which is easy) and pass to stage $s+1$, setting $\rho_{s+1} = r^t$ and selecting, as the first iterate of the new stage, any point known to belong to $K[\rho]$ (e.g., the origin, or $\bar{x}_t$). The first iterate of the first stage is 0.

The description of the algorithm is complete.

The complexity properties of the algorithm are given by the following proposition.

Theorem 2. *When solving a PO problem (6), (7) by the outlined algorithm,*

(i) *the algorithm terminates with an ϵ-solution, as defined in Section 2 (cf. (8));*

(ii) *The number N_s of steps at every stage s of the method admits the bound*

$$N_s \le \max\left[6, \frac{72\rho_*^2 L_f}{\epsilon} + 3\right].$$

(iii) *The number of stages before termination does not exceed the quantity*

$$\max\left[1.2\ln\left(\frac{f(0) + \frac{1}{2}L_f\rho_*^2}{\epsilon^2}\right) + 2.4, 3\right].$$

[4] The iterates x_t, same as other indexed by t quantities participating in the description of the algorithm, in fact depend on both t and the stage number s. To avoid cumbersome notation when speaking about a particular stage, we suppress s in the notation.

5 Conditional Gradient algorithm for Composite Optimization

In this section, we present a modification of the CndG algorithm capable to solve composite minimization problem (10). We assume in the sequel that $\|\cdot\|, K$ are represented by an LO oracle for the set $\{x \in K : \|x\| \le 1\}$, and f is given by a first order oracle. In order to apply CndG to the composite optimization problem (10), we make the assumption as follows:

> **Assumption A:** There exists $D < \infty$ such that $\kappa r + f(x) \le f(0)$ together with $\|x\| \le r$, $x \in K$, imply that $r \le D$.

We define D_* as the minimal value of D satisfying Assumption A, and assume that *we have at our disposal a finite upper bound D^+ on D_*.* An important property of the algorithm we are about to develop is that its efficiency estimate depends on the induced by problem's data quantity D_*, and is independent of our a priori upper bound D^+ on this quantity, see Theorem 3 below.

The algorithm. We are about to present an algorithm for solving (10). Let $E^+ = E \times \mathbf{R}$, and $K^+ = \{[x; r] : x \in K, \|x\| \le r\}$. From now on, for a point $z = [x; r] \in E^+$ we set $x(z) = x$ and $r(z) = r$. Given $z = [x; r] \in K^+$, let us consider the segment

$$\Delta(z) = \{\rho[x[f'(x)]; 1] : 0 \le \rho \le D^+\}.$$

and the linear form

$$\zeta = [\xi; \tau] \to \langle f'(x), \xi\rangle + \kappa\tau = \langle F'(z), \zeta\rangle$$

Observe that by Lemma 1, for every $0 \le \rho \le D^+$, the minimum of this form on $K^+[\rho] = \{[x; r] \in E^+, x \in K, \|x\| \le r \le \rho\}$ is attained at a point of $\Delta(z)$ (either at $[\rho x[f'(x)]; \rho]$ or at the origin). A generic Conditional Gradient algorithm for composite optimization (COCndG) is a recurrence which builds the points $z_t = [x_t; r_t] \in K^+$, $t = 1, 2, ...$, in such a way that

$$z_1 = 0;\ F(z_{t+1}) \le \min_z\{F(z) : z \in \mathrm{Conv}\,(\Delta(z_t) \cup \{z_t\})\},\ \ t = 1, 2, ... \tag{24}$$

Let $z_* = [x_*; r_*]$ be an optimal solution to (10) (which under Assumption A clearly exists), and let $F_* = F(z_*)$ (i.e., F_* is nothing but Opt, see (9)).

Theorem 3. *A generic COCndG algorithm (24) maintains the inclusions $z_t \in K^+$ and is a descent algorithm: $F(z_{t+1}) \le F(z_t)$ for all t. Besides this, we have*

$$F(z_t) - F_* \le \frac{8L_f D_*^2}{t + 14},\ t = 2, 3, ... \tag{25}$$

COCndG with memory. The simplest implementation of a generic COCndG algorithm is given by the recurrence

$$z_1 = 0;\ z_{t+1} \equiv [x_{t+1}; r_{t+1}] \in \mathrm{Argmin}_z\,\{F(z) : z \in \mathrm{Conv}\,(\Delta(z_t) \cup \{z_t\})\},\ \ t = 1, 2, ... \tag{26}$$

Denoting $\widehat{z}_\tau := D^+[x[f'(x_\tau)]; 1]$, the recurrence can be written

$$\begin{array}{l} z_{t+1} = \lambda_t \widehat{z}_t + \mu_t z_t, \text{ where} \\ (\lambda_t, \mu_t) \in \mathrm{Argmin}_{\lambda,\mu} \left\{F(\lambda\widehat{z}_t + \mu z_t) : \lambda + \mu \le 1,\ \lambda \ge 0, \mu \ge 0\right\}. \end{array} \tag{27}$$

As for the CndG algorithm in section 3, the recurrence (26) admits a version with memory COCndGM still obeying (24) and thus sartisfying the conclusion of Theorem 3. Specifically, assume that we already have built t iterates $z_\tau = [x_\tau; r_\tau] \in K^+$, $1 \le \tau \le t$, with $z_1 = 0$, along with the gradients $f'(x_\tau)$ and the points $x[f'(x_\tau)]$. Then we have at our disposal a number of points from K^+, namely, the iterates z_τ, $\tau \le t$, and the points $\widehat{z}_\tau = D^+[x[f'(x_\tau)]; 1]$. Let us select a subset Z_t of the set $\{z_\tau, \widehat{z}_\tau, 1 \le \tau \le t\}$, with the only restriction that Z_t contains the points $z_t, \widehat{z}_t$, and set

$$z_{t+1} \in \operatorname*{Argmin}_{z \in \mathcal{C}_t} F(z), \quad \mathcal{C}_t = \mathrm{Conv}\{\{0\} \cup Z_t\}\}. \tag{28}$$

Since $z_t, \widehat{z}_t \in Z_t$, we have $\mathrm{Conv}\,(\Delta(z_t) \cup \{z_t\})\} \subset \mathcal{C}_t$, whence the procedure we have outlined is an implementation of generic COCndG algorithm. Note that the basic COCndG algorithm is the particular case of the COCndGM corresponding to the case where $Z_t = \{z_t, \widehat{z}_t\}$ for all t. The discussion of implementability of CndGM in section 3 fully applies to COCndGM.

Let us outline several options which can be implemented in COCndGM; while preserving the theoretical efficiency estimates stated in Theorem 3 they can improve the practical performance of the algorithm. For the sake of definiteness, let us focus on the case of quadratic f: $f(x) = \|\mathcal{A}x - b\|_2^2$, with $\mathrm{Ker}\mathcal{A} = \{0\}$; extensions to a more general case are straightforward.

A. We lose nothing (and potentially gain) when extending $\mathcal{C}_t$ in (28) to the conic hull

$$\mathcal{C}_t^+ = \{w = \sum_{\zeta \in Z_t} \lambda_\zeta \zeta : \ \lambda_\zeta \ge 0, \ \zeta \in Z_t\}$$

of Z_t. When $K = E$, we can go further and replace (28) with

$$z_{t+1} \in \operatorname*{Argmin}_{z=[x;r],\lambda} \left\{ f(x) + \kappa r : \ x = \sum_{\zeta=[\eta;\rho] \in Z_t} \lambda_\zeta \eta, \ r \ge \sum_{\zeta=[\eta;\rho] \in Z_t} |\lambda_\zeta| \rho \right\}. \tag{29}$$

Note that the preceding "conic case" is obtained from (29) by adding to the constraints of the right hand side problem the inequalities $\lambda_\zeta \ge 0, \zeta \in Z_t$. Finally, when $\|\cdot\|$ is easy to compute, we can improve (29) to

$$\begin{array}{c} z_{t+1} = \left[\sum_{\zeta=[\eta;\rho] \in Z_t} \lambda_\zeta^* \eta; \left\|\sum_{\zeta=[\eta;\rho] \in Z_t} \lambda_\zeta^* \eta\right\|\right], \\ \lambda^* \in \mathrm{Argmin}_{\{\lambda_\zeta, \zeta \in Z_t\}} \left\{ f\left(\sum_{\zeta=[\eta;\rho] \in Z_t} \lambda_\zeta \eta\right) + \kappa \sum_{\zeta=[\eta;\rho] \in Z_t} |\lambda_\zeta| \rho \right\} \end{array} \tag{30}$$

(the definition of λ^* assumes that $K = E$, otherwise the constraints of the problem specifying λ^* should be augmented by the inequalities $\lambda_\zeta \ge 0, \zeta \in Z_t$).

B. In the case of quadratic f and moderate cardinality of Z_t, optimization problems arising in (29) (with or without added constraints $\lambda_\zeta \ge 0$) are explicitly given low-dimensional "nearly quadratic" convex problems which can be solved to high accuracy "in no time" by interior point solvers. With this in mind, we could solve these problems for the given value of the penalty parameter κ *and also for several other values of the parameter*. Thus, at every iteration we get feasible approximate solution to several instances of (9) for different values of the penalty parameter. Assume that we keep in memory, for every value of the penalty parameter in question, the best, in terms of the respective objective, of the related approximate solutions found so far. Then upon termination we will have at our disposal, along with

the feasible approximate solution associated with the given value of the penalty parameter, provably obeying the efficiency estimates of Theorem 3, a set of feasible approximate solutions to the instances of (9) corresponding to other values of the penalty.

C. In the above description, Z_t was assumed to be a subset of the set $Z^t = \{z_\tau = [x_\tau; r_\tau], \widehat{z}_\tau,\ 1 \le \tau \le t\}$ containing z_t and $\widehat{z}_t$. Under the latter restriction, we lose nothing when allowing for Z_t to contain points from $K^+ \backslash Z^t$ as well. For instance, when $K = E$ and $\|\cdot\|$ is easy to compute, we can add to Z_t the point $z'_t = [f'(x_t); \|f'(x_t)\|]$. Assume, e.g., that we fix in advance the cardinality $M \ge 3$ of Z_t and define Z_t as follows: to get Z_t from Z_{t-1}, we eliminate from the latter set several (the less, the better) points to get a set of cardinality $\le M-3$, and then add to the resulting set the points z_t, $\widehat{z}_t$ and z'_t. Eliminating the points according to the rule "first in – first out," the projection of the feasible set of the optimization problem in (30) onto the space of x-variables will be a linear subspace of E containing, starting with step $t = M$, at least $\lfloor M/3 \rfloor$ (here $\lfloor a \rfloor$ stands for the largest integer not larger than a) of gradients of f taken at the latest iterates, so that the method, modulo the influence of the penalty term, becomes a "truncated" version of the Conjugate Gradient algorithm for quadratic minimization. Due to nice convergence properties of Conjugate Gradient in the quadratic case, one can hope that a modification of this type will improve significantly the practical performance of COCndGM.

6 Application examples

In this section, we detail how the proposed conditional gradient algorithms apply to several examples. In particular, we detail the corresponding LO oracles, and how one could implement these oracles efficiently.

6.1 Regularization by nuclear/trace norm

The first example where the proposed algorithms seem to be more attractive than the proximal methods are large-scale problems (5), (9) on the space of $p \times q$ matrices $E = \mathbf{R}^{p\times q}$ associated with the nuclear norm $\|\sigma(x)\|_1$ of a matrix x, where $\sigma(x) = [\sigma_1(x); ...; \sigma_{\min[p,q]}(x)]$ is the vector of singular values of a $p \times q$ matrix x. Problems of this type with $K = E$ arise in various versions of matrix completion, where the goal is to recover a matrix x from its noisy linear image $y = \mathcal{A}x + \xi$, so that $f = \phi(\mathcal{A}x - y)$, with some smooth and convex discrepancy measure $\phi(\cdot)$, most notably, $\phi(z) = \frac{1}{2}\|z\|_2^2$. In this case, $\|\cdot\|$ minimization/penalization is aimed at getting a recovery of low rank ([31, 3, 4, 9, 15, 26, 27, 33, 20, 29] and references therein). Another series of applications relates to the case when $E = \mathbf{S}^p$ is the space of symmetric $p \times p$ matrices, and $K = \mathbf{S}^p_+$ is the cone of positive semidefinite matrices, with f and ϕ as above; this setup corresponds to the situation when one wants to recover a covariance (and thus positive semidefinite symmetric) matrix from experimental data. Restricted from $\mathbf{R}^{p\times p}$ onto $\mathbf{S}^p$, the nuclear norm becomes the trace norm $\|\lambda(x)\|_1$, where $\lambda(x) \in \mathbf{R}^p$ is the vector of eigenvalues of a symmetric $p \times p$ matrix x, and regularization by this norm is, as above, aimed at building a low rank recovery.

With the nuclear (or trace) norm in the role of $\|\cdot\|$, all known proximal algorithms require, at least in theory, computing at every iteration the complete singular value decomposition of $p \times q$ matrix x (resp., complete eigenvalue decomposition of a symmetric $p \times p$ matrix x), which for large p, q may become prohibitively time consuming. In contrast to this, with $K = E$ and $\|\cdot\| = \|\sigma(\cdot)\|_1$, LO oracle for $(K, \|\cdot\| = \|\sigma(\cdot)\|_1)$ only requires computing the leading right singular vector e of a $p\times q$

matrix η (i.e., the leading eigenvector of $\eta^T\eta$): $x[\eta] = -\bar{f}\bar{e}^T$, where $\bar{e} = e/\|e\|_2$ and $\bar{f} = \eta e/\|\eta e\|_2$ for nonzero η and $\bar{f} = 0$, $\bar{e} = 0$ when $\eta = 0$. Computing the leading singular vector of a large matrix is, in most cases, much cheaper than computing the complete eigenvalue decomposition of the matrix. Similarly, in the case of $E = \mathbf{S}^p$, $K = \mathbf{S}^p_+$ and the trace norm in the role of $\|\cdot\|$, LO oracle requires computing the leading eigenvector e of a matrix $\eta \in \mathbf{S}^p$: $x[-\eta] = \bar{e}\bar{e}^T$, where $\bar{e} = 0$ when $e^T\eta e \geq 0$, and $\bar{e} = e/\|e\|_2$ otherwise. Here again, for a large symmetric $p \times p$ matrix, the required computation usually is much easier than computing the complete eigenvalue decomposition of such a matrix. As a result, in the situations under consideration, algorithms based on the LO oracle remain "practically implementable" in an essentially larger range of problem sizes than proximal methods.

An additional attractive property of the CndG algorithms we have described stems from the fact that *since in the situations in question the matrices $x[\eta]$ are of rank 1, t-th approximate solution x_t yielded by the CndG algorithms for composite minimization from Section 5 is of rank at most t. Similar statement holds true for t-th approximate solution x_t built at a stage of a CndG algorithm for parametric optimization from Section 3, provided that the first iterate at every stage is the zero matrix.*[5].

6.2 Regularization by Total Variation

Given integer $n \geq 2$, consider the linear space $M^n := \mathbf{R}^{n\times n}$. We interpret elements x of M^n as *images* – real-valued functions $x(i,j)$ on the $n \times n$ grid $\Gamma_{n,n} = \{[i;j]) \in \mathbf{Z}^2 : 0 \leq i,j < n\}$. The (anisotropic) Total Variation (TV) of an image x is the ℓ_1-norm of its (discrete) gradient field $(\nabla_i x(\cdot), \nabla_j x(\cdot))$:

$$\begin{array}{rcl} & \mathrm{TV}(x) = \|\nabla_i x\|_1 + \|\nabla_j x\|_1, & \\ \nabla_i x(i,j) & = & x_(i+1,j) - x(i,j) : \Gamma_{n-1,n} := \{[i;j] \in \mathbf{Z}^2 : 0 \leq i < n-1, 0 \leq j < n\}, \\ \nabla_j x(i,j) & = & x_(i,j+1) - x(i,j) : \Gamma_{n,n-1} := \{[i;j] \in \mathbf{Z}^2 : 0 \leq i < n, 0 \leq j < n-1\} \end{array}$$

Note that $\mathrm{TV}(\cdot)$ is a norm on the subspace M^n_0 of M^n comprised of *zero mean images* x (those with $\sum_{i,j} x(i,j) = 0$) and vanishes on the orthogonal complement to M^n_0, comprised of constant images.

Originating from the celebrated paper [28] and extremely popular *Total Variation-based image reconstruction* in its basic version recovers an image x from its noisy observation $b = \mathcal{A}x + \xi$ by solving problems (5) or (9) with $K = E = M^n$, $f(x) = \phi(\mathcal{A}x - b)$ and the seminorm $\mathrm{TV}(\cdot)$ in the role of $\|\cdot\|$. In the sequel, we focus on the versions of these problems where $K = E = M^n$ is replaced with $K = E = M^n_0$, thus bringing the TV-regularized problems into our framework. This restriction is basically harmless; for example, in the most popular case of $f(x) = \frac{1}{2}\|\mathcal{A}x - b\|_2^2$ reduction to the case of $x \in M^n_0$ is immediate – it suffices to replace $(\mathcal{A}, b)$ with $(P\mathcal{A}, Pb)$, where P is the orthoprojector onto the orthogonal complement to the one-dimensional subspace spanned by $\mathcal{A}\mathbf{e}$, where $\mathbf{e}$ is the all-ones image[6]. Now, large scale problems (5), (9) with $K = E = M^n_0$ and $\mathrm{TV}(\cdot)$ in the role of $\|\cdot\|$ are difficult to solve by proximal algorithms. Indeed, in the situation in question a proximal algorithm would require at every iteration either minimizing function of the form $\mathrm{TV}(x) + \langle e, x\rangle + \omega(x)$ over

[5] this property is an immediate corollary of the fact that in the situation in question, by description of the algorithms x_t is a convex combination of t points of the form $x[\cdot]$.

[6] When f is more complicated, optimal adjustment of the mean t of the image reduces by bisection in t to solving small series of problems of the same structure as (5), (9) where the mean of the image x is fixed and, consequently, the problems reduce to those with $x \in M^n_0$ by shifting b.

the entire E, or minimizing function of the form $\langle e, x\rangle + \omega(x)$ on a TV-ball[7], where $\omega(x)$ is albeit simple, but *nonlinear* convex function (e.g., $\|x\|_2^2$, or $\|\nabla_i x\|_2^2 + \|\nabla_j x\|_2^2$). Auxiliary problems of this type seem to be difficult in the large scale case, especially taking into account that when running a proximal algorithm we need to solve at least tens, and more realistically – hundreds of them[8]. In contrast to this, a LO oracle for the unit ball $\mathcal{TV} = \{x \in M_0^n : \mathrm{TV}(x) \le 1\}$ of the TV norm is relatively cheap computationally – it reduces to solving a specific maximum flow problem. It should be mentioned here that the relation between flow problems and TV-based denoising (problem (9) with $\mathcal{A} = I$) is well known and is utilized in many algorithms, see [9] and references therein. While we have no doubt that the simple fact stated Lemma 2 below is well-known, for reader convenience we present here in detail the reduction mechanism.

Consider the network (the oriented graph) G with n^2 nodes $[i; j] \in \Gamma_{n,n}$ and $2n(n-1)$ arcs as follows: the first $n(n-1)$ arcs are of the form $([i+1; j], [i; j])$, $0 \le i < n-1$, $0 \le j < n$, the next $n(n-1)$ arcs are $([i; j+1], [i; j])$, $0 \le i < n$, $0 \le j < n-1$, and the remaining $2n(n-1)$ arcs (let us call them *backward* arcs) are the inverses of the just defined $2n(n-1)$ *forward* arcs. Let $\mathcal{E}$ be the set of arcs of our network, and let us equip all the arcs with unit capacities. Let us treat vectors from $E = M_0^n$ as vectors of external supplies for our network; note that the entries of these vectors sum to zero, as required from external supply. Now, given a nonzero vector $\eta \in M_0^n$, let us consider the network flow problem where we seek for the largest multiple $s\eta$ of η which, considered as the vector of external supplies in our network, results in a feasible capacitated network flow problem. The problem in question reads

$$s_* = \max_{s,r} \{s : Pr = s\eta,\ 0 \le r \le \mathbf{e}\}, \tag{31}$$

where P is the incidence matrix of our network[9] and $\mathbf{e}$ is the all-ones vector. Now, problem (31) clearly is feasible, and its feasible set is bounded due to $\eta \ne 0$, so that the problem is solvable. Due to its network structure, this LP program can be solved reasonably fast even in the large scale case (say, when $n = 512$ or $n = 1024$, which already is of interest for actual imaging). Further, an intelligent network flow solver as applied to (31) will return not only the optimal $s = s_*$ and the corresponding flow, but also the dual information, in particular, the optimal vector z of Lagrange multipliers for the linear equality constraints $Pr - s\eta = 0$. Let $\bar{z}$ be obtained by subtracting from the entries of z their mean; since the entries of z are indexed by the nodes, $\bar{z}$ can be naturally interpreted as a zero mean image. It turns out that this image is nonzero, and the vector $x[\eta] = -\bar{z}/\mathrm{TV}(\bar{z})$ is nothing than a desired minimizer of $\langle \eta, \cdot\rangle$ on $\mathcal{TV}$:

Lemma 2. *Let η be a nonzero image with zero mean. Then (31) is solvable with positive optimal value, and the image $x[\eta]$, as defined above, is well defined and is a maximizer of $\langle \eta, \cdot\rangle$ on $\mathcal{TV}$.*

[7] which one of these two options takes place depends on the type of the algorithm.

[8] On a closest inspection, "complex geometry" of the TV-norm stems from the fact that after parameterizing a zero mean image by its discrete gradient field and treating this field ($g = \nabla_i x, h = \nabla_j x$) as our new design variable, the unit ball of the TV-norm becomes the intersection of a simple set in the space of pairs $(g, h) \in F = \mathbf{R}^{(n-1)\times n} \times \mathbf{R}^{n\times(n-1)}$ (the ℓ_1 ball Δ given by $\|g\|_1 + \|h\|_1 \le 1$) with a linear subspace P of F comprised of *potential* vector fields (f, g) – those which indeed are discrete gradient fields of images. Both dimension and codimension of P are of order of n^2, which makes it difficult to minimize over $\Delta \cap P$ *nonlinear*, even simple, convex functions, which is exactly what is needed in proximal methods.

[9] that is, the rows of P are indexed by the nodes, the columns are indexed by the arcs, and in the column indexed by an arc γ there are exactly two nonzero entries: entry 1 in the row indexed by the starting node of γ, and entry -1 in the row indexed by the terminal node of γ.

Bounding L_f. When applying CndG algorithms to the TV-based problems (5), (9) with $E = M_0^n$ and $f(x) = \phi(\mathcal{A}x - b)$, the efficiency estimates depend linearly on the associated quantity L_f, which, in turn, is readily given by the norm $\|\mathcal{A}\|_{\mathrm{TV}(\cdot),\pi(\cdot)}$ of the mapping $x \mapsto \mathcal{A}x$, see the end of Section 2. Observe that in typical applications $\mathcal{A}$ is a simple operator (e.g., the discrete convolution), so that when restricting ourselves to the case when $\pi(\cdot)$ is $\|\cdot\|_2$ (quadratic fit), it is easy to find a tight upper bound on $\|\mathcal{A}\|_{\|\cdot\|_2,\|\cdot\|_2}$. To convert this bound into an upper bound on $\|\mathcal{A}\|_{\mathrm{TV}(\cdot),\|\cdot\|_2}$, we need to estimate the quantity

$$Q_n = \max_x\{\|x\|_2 : x \in M_0^n, \mathrm{TV}(x) \le 1\}.$$

Bounding Q_n is not a completely trivial question, and the answer is as follows:

Proposition 1. *Q_n is nearly constant, specifically, $Q_n \le O(1)\sqrt{\ln(n)}$ with a properly selected absolute constant $O(1)$.*

Note that the result of Proposition 1 is in sharp contrast with one-dimensional case, where the natural analogy of Q_n grows with n as $\sqrt{n}$. We do not know whether it is possible to replace in Proposition 1 $O(1)\sqrt{\ln(n)}$ with $O(1)$, as suggested by Sobolev's inequalities[10]. Note that on inspection of the proof, Proposition extends to the case of d-dimensional, $d > 2$, images with zero mean, in which case $Q_n \le C(d)$ with appropriately chosen $C(d)$.

7 Numerical examples

We present here some very preliminary simulation results.

7.1 CndG for parametric optimization: sparse matrix completion problem

The goal of the first series of our experiments is to illustrate how the performance and requirements of CndG algorithm for parametric optimization, when applied to the matrix completion problem [4], scale with problem size. Specifically, we apply the algorithm of Section 4 to the problem of nuclear norm minimization

$$\min \|\sigma(x)\|_1, \ \text{ subject to } \sum_{(i,j)\in\Omega} (y_{ij} - x_{ij})^2 \le \delta, \tag{32}$$

where $\sigma(x)$ is the singular spectrum of a $p \times q$ matrix x. In our experiments, the set Ω of observed entries $(i, j) \in \{1, ..., p\} \times \{1, ..., q\}$ of cardinality $m \ll pq$ was selected at random.

Note that the the implementation of the CndGM is especially simple for the problem (32) – at each method's iteration it requires solving a simple quadratic problem with dimension of the decision variable which does not exceed the iteration count. This allows to implement efficiently the "full memory" version of CndGM (CndG algorithms with memory) (21), (22), in which the set X_t contains x_t and all the points x_τ^+ for $1 \le \tau \le t$.

We compare the performance of CndGM algorithms and of a "memoryless" version of the CndG. To this end we have conducted the following experiment:

[10] From the Sobolev embedding theorem it follows that for a smooth function $f(x, y)$ on the unit square one has $\|f\|_{L_2} \le O(1)\|\nabla f\|_1$, $\|\nabla f\|_1 := \|f'_x\|_1 + \|f'_y\|_1$, provided that f has zero mean. Denoting by f^n the restriction of the function onto a $n \times n$ regular grid in the square, we conclude that $\|f^n\|_2/\mathrm{TV}(f^n) \to \|f\|_{L_2}/\|\nabla f\|_1 \le O(1)$ as $n \to \infty$. Note that the convergence in question takes place only in the 2-dimensional case.

1. For matrix sizes $p, q \in [1, 2, 4, 8, 16, 32] \times 10^3$ we generate $n = 10$ sparse $p \times q$ matrices y with density $d = 0.1$ of non-vanishing entries as follows: we generate $p \times r$ matrix U and $q \times r$ matrix V with independent Gaussian entries $u_{ij} \sim \mathcal{N}(0, m^{-1})$, $v_{ij} \sim \mathcal{N}(0, n^{-1})$, and a $r \times r$ diagonal matrix $D = \mathrm{diag}[d_1, ..., d_r]$ with d_i drawn independently from a uniform distribution on $[0, 1]$. The non-vanishing entries of the sparse observation matrix y are obtained by sampling at random with probability d the entries of $x^* = UDV^T$, so that for every i, j, y_{ij} is, independently over i, j, set to x^*_{ij} with probability d and to 0 with probability $1 - d$. Thus, the number of non-vanishing entries of y is approximately $m = dpq$. This procedure results in $m \sim 10^5$ for the smallest matrices y (1000×1000), and in $m \sim 10^8$ for the largest matrices (32000×32000).
2. We apply to parametric optimization problem (32) MATLAB implementations of the CndGM with memory parameter $M = 1$ ("memoryless" CndG), CndGM with $M = 5$ and full memory CndGM. The parameter δ of (32) is chosen to be $\delta = 0.001\|y\|_{\mathrm{f}}^2$ (here $\|y\|_{\mathrm{f}} = \left(\sum_{i,j} y_{ij}^2\right)^{1/2}$ stands for the Frobenius norm of y). The optimization algorithm is tuned to the *relative accuracy* $\varepsilon = 1/4$, what means that it outputs an ϵ-solution $\widehat{x}$ to (32), in the sense of (8), with absolute accuracy $\epsilon = \delta\varepsilon$.

For each algorithm (memoryless CndG, CndGM with memory $M = 5$ and full memory CndGM) we present in table 1 the average, over algorithm's runs on the (common for all algorithms) sample of $n = 10$ matrices y we have generated, 1) total number of iterations N_{it} necessary to produce an ϵ-solution (it upper-bounds the rank of the resulting ϵ-solutuion), 2) CPU time in seconds T_{cpu} and 3) MATLAB memory usage in megabytes S_{mem}. This experiment was conducted on a Dell Latitude 6430 laptop equipped with Intel Core i7-3720QM CPU@2.60GHz and 16GB of RAM. Because of high memory requirements in our implementation of the full memory CndGM, this method was unable to complete the computation for the two largest matrix sizes.

We can make the following observation regarding the results summarized in table 1: CndG algorithm with memory consistently outperforms the standard – memoryless – version of CndG. The full memory CndGM requires the smallest number of iteration to produce an ϵ-solution, which is of the smallest rank, as a result. On the other hand, the memory requirements of the full memory CndGM become prohibitive (at least, for the computer we used for this experiment and MATLAB implementation of the memory heap) for large matrices. On the other hand, a CndGM with memory $M = 5$ appears to be a reasonable compromise in terms of numerical efficiency and memory demand.

7.2 CndG for composite optimization: multi-class classification with nuclear-norm regularization

We present here an empirical study of the CndG algorithm for composite optimization as applied to the machine learning problem of multi-class classification with nuclear-norm penalty. A brief description of the multi-class classification problem is as follows: we observe N "feature vectors" $\xi_i \in \mathbf{R}^q$, each belonging to exactly one of p classes $C_1, ..., C_p$. Each ξ_i is augmented by its label $y_i \in \{1, ..., p\}$ indicating to which class ξ_i belongs. Our goal is to build a classifier capable to predict the class to which a new feature vector ξ belongs. This classifier is given by a $p \times q$ matrix x according to the following rule: given ξ, we compute the p-dimensional vector $x\xi$ and take, as the guessed class of ξ, the index of the largest entry in this vector.

In some cases (see [6, 10]), when, for instance, one is dealing with a large number of classes,

Matrix size	Memory-less CndG			CndGM with memory $M=5$			Full memory CndG		
$p \times q$	$N_{\rm it}$	$T_{\rm cpu}$	$S_{\rm mem}$	$N_{\rm it}$	$T_{\rm cpu}$	$S_{\rm mem}$	$N_{\rm it}$	$T_{\rm cpu}$	$S_{\rm mem}$
1000×1000	271.6	9.35	17.11	149.7	5.01	17.63	78.4	4.71	78.98
1000×2000	292.1	12.14	31.67	162.8	7.76	32.57	93.5	10.89	156.22
2000×2000	246.8	17.01	54.45	139.1	11.19	61.57	71.9	13.31	248.13
2000×4000	259.3	33.94	105.09	152.3	24.50	120.22	57.7	25.54	410.02
4000×4000	321.8	79.20	207.26	162.9	50.59	235.59	74.6	93.22	1014.7
4000×8000	360.1	169.8	399.16	147.3	88.81	464.68	63.3	135.6	1766.4
8000×8000	323.4	302.8	754.46	111.8	134.1	905.98	53.6	191.3	3061.5
8000×16000	324.1	614.3	1485.6	118.2	286.5	1800.7	50.5	329.4	5826.7
16000×16000	258.7	995.4	2898.5	99.7	495.5	3577.8	50.8	595.2	11696
16000×32000	276.7	2572	5721.7	70.3	859.2	7109.0	NA	NA	NA
32000×32000	305.4	5028	11352	57.6	2541	14186	NA	NA	NA

Table 1: memoryless CndG vs. CndGM with memory $M = 5$ vs. full memory CndGM. $N_{\rm it}$: total number of method iterations; $T_{\rm cpu}$: CPU usage (sec), and $S_{\rm mem}$: memory usage (MB) reported by MATLAB.

there are good reasons "to train the classifier" — to specify x given the training sample (ξ_i, y_i), $1 \le i \le N$ — as the optimal solution to the nuclear norm penalized minimization problem

$$\mathrm{Opt}(\kappa) = \min_{x \in \mathbf{R}^{p\times q}} \quad F_\kappa(x) := \overbrace{\frac{1}{N}\sum_{i=1}^{N} \log\left\{\sum_{\ell=1}^{q} \exp\left((x_\ell^T - x_{y_i}^T)\xi_i\right)\right\}}^{f(x)} + \kappa\|\sigma(x)\|_1, \tag{33}$$

where x_ℓ^T is the ℓ-th row in x.

Below, we report on some experiments with this problem. Our goal was to compare two versions of CndG for composite minimization: the memoryless version defined in (24) and the version with memory defined in (28). To solve the corresponding sub-problems, we used the Center of Gravity method in the case of (24) and the Ellipsoid method in the case of (28) [22, 21]. In the version with memory we set $M = 5$, as it appeared to be the best option from empirical evidence. We have considered the following datasets:

1. **Simulated data**: for matrix of sizes $p, q \in 10^3 \times \{2^s\}_{s=1}^4$, we generate random matrices $x_\star = USV$, with $p \times p$ factor U, $q \times q$ factor V, and diagonal $p \times q$ factor S with random entries sampled, independently of each other, from $\mathcal{N}(0, p^{-1})$ (for U), $\mathcal{N}(0, q^{-1})$ (for V), and the uniform distribution on $[0, 1]$ (for diagonal entries in S). We use $N = 20q$, with the feature vectors $\xi_1, ..., \xi_N$ sampled, independently of each other, from the distribution $\mathcal{N}(0, I_q)$, and their labels y_i being the indexes of the largest entries in the vectors $x_\star\xi_i + \epsilon_i$, where $\epsilon_i \in \mathbf{R}^p$ were sampled, independently of each other and of $\xi_1, ..., \xi_N$, from $\mathcal{N}(0, \frac{1}{2}I_p)$. The regularization parameter κ is set to $10^{-3}\mathrm{Tr}(x_\star x_\star^T)$.
2. **Real-world data**: we follow a setting similar to [10]. We consider the Pascal ILSVRC2010 ImageNet dataset and focus on the "Vertebrate-craniate" subset, yielding 1043 classes, with 20 examples per class. The goal here is to train a multi-class classifier in order to be able to predict the class of each image (example) of the dataset. Each example is converted to a 65536-dimensional feature vector of unit ℓ_1-norm using state-of-the-art visual descriptors known as Fisher vector representation [10]. To summarize, we have $p = 1043$, $q = 65536$,

Matrix size	Memory-less CndG			CndGM with memory $M = 5$		
$p \times q$	$N_{\rm it}$	$T_{\rm cpu}$	$S_{\rm mem}$	$N_{\rm it}$	$T_{\rm cpu}$	$S_{\rm mem}$
2000×2000	172.9	349.7	134.4	99.70	125.1	174.1
4000×4000	153.4	1035	541.8	88.2	575.2	704.1
8000×8000	195.3	2755	2169	120.4	1284	2819
16000×16000	230.2	6585	8901	134.3	3413	11550
32000×32000	271.4	26370	30300	140.4	17340	30500
1043×65536	183	2101	2087	111	925.34	2709

Table 2: memoryless CndG vs. CndGM with memory $M = 5$. $N_{\rm it}$: total number of method iterations; $T_{\rm cpu}$: CPU usage (sec) reported by MATLAB.

$N = 20860$. We set the regularization parameter to $\kappa = 10^{-4}$, which was found to result in the best predictive performance as estimated by cross-validation, a standard procedure to set the hyper parameters in machine learning [11].

In both sets of experiments, the computations are terminated when the "ϵ-optimality conditions"

$$\begin{array}{rcl} \|\sigma(f'(x_t))\|_\infty & \leq & \kappa + \epsilon \\ \langle f'(x_t), x_t\rangle + \kappa\|\sigma(x_t)\|_1 & \leq & \epsilon\|\sigma(x_t)\|_1 \end{array} \tag{34}$$

were met, where $\|\sigma(\cdot)\|_\infty$ denotes the usual operator norm (the largest singular value). These conditions admit transparent interpretation as follows. For every $\bar{x}$, the function

$$\phi_\kappa(x) = f(\bar{x}) + \langle f'(\bar{x}), x - \bar{x}\rangle + \kappa\|\sigma(x)\|_1$$

underestimates $F_\kappa(x)$, see (33), whence $\mathrm{Opt}(\kappa') \geq f(\bar{x}) - \langle f'(\bar{x}), \bar{x}\rangle$ whenever $\kappa' \geq \|\sigma(f'(\bar{x}))\|_\infty$. Thus, whenever $\bar{x} = x_t$ satisfies the first relation in (34), we have $\mathrm{Opt}(\kappa + \epsilon) \geq f(x_t) - \langle f'(x_t), x_t\rangle$, whence

$$F_\kappa(x_t) - \mathrm{Opt}(\kappa + \epsilon) \leq \langle f'(x_t), x_t\rangle + \kappa\|\sigma(x_t)\|_1.$$

We see that (34) ensures that $F_\kappa(x_t) - \mathrm{Opt}(\kappa + \epsilon) \leq \epsilon\ \|\sigma(x_t)\|_1$, which, for small ϵ, is a reasonable substitute for the actually desired termination when $F_\kappa(x_t) - \mathrm{Opt}(\kappa)$ becomes small. In our experiments, we use $\epsilon = 0.001$.

In table 2 for each algorithm (memoryless CndG, CndGM with memory $M = 5$) we present the average, over 20 collections of simulated data coming from 20 realizations of $x_\star$, of: 1) total number of iterations $N_{\rm it}$ necessary to produce an ϵ-solution, 2) CPU time in seconds $T_{\rm cpu}$. The last row of the table corresponds to the real-world data. Experiments were conducted on a Dell R905 server equipped with four six-core AMD Opteron 2.80GHz CPUs and 64GB of RAM. A maximum of 32GB of RAM was used for the computations.

We draw the following conclusions from table 1: CndG algorithm with memory routinely outperforms the standard – memoryless – version of CndG. However, there is a trade-off between the algorithm progress at each iteration and the computational load of each iteration. Note that, for large M, solving the sub-problem (28) can be challenging.

7.3 CndG for composite optimization: TV-regularized image reconstruction

Here we report on experiments with COCndGM as applied to TV-regularized image reconstruction. Our problem of interest is of the form (9) with quadratic f, namely, the problem

$$\min_{x\in M_0^n} \phi_\kappa(x) := \underbrace{\frac{1}{2}\|P\mathcal{A}x - Pb\|_2^2}_{f(x)} + \kappa \mathrm{TV}(x); \tag{35}$$

for notation, see section 6.2.

Test problems. In our experiments, the mapping $x \mapsto \mathcal{A}x$ is defined as follows: we zero-pad x to extend it from $\Gamma_{n,n}$ to get a finitely supported function on $\mathbf{Z}^2$, then convolve this function with a finitely supported kernel $\alpha(\cdot)$, and restrict the result onto $\Gamma_{n,n}$. The observations $b \in M^n$ were generated at random according to

$$b_{ij} = (\mathcal{A}x)_{ij} + \sigma\|x\|_\infty \xi_{ij},\ \xi_{ij} \sim \mathcal{N}(0,1),\ 1 \le i,j \le n, \tag{36}$$

with mutually independent ξ_{ij}. The relative noise intensity $\sigma > 0$, same as the convolution kernel $\alpha(\cdot)$, are parameters of the setup of an experiment.

The algorithm. We used the COCndG with memory, described in section 5; we implemented the options listed in **A** – **C** at the end of the section. Specifically,

1. We use the updating rule (30) with Z_t evolving in time exactly as explained in item **C**: the set Z_t is obtained from Z_{t-1} by adding the points $z_t = [x_t; \mathrm{TV}(x_t)]$, $\widehat{z}_t = [x[\nabla f(x_t)]; 1]$ and $z'_t = [\nabla f(x_t); \mathrm{TV}(\nabla f(x_t))]$, and deleting from the resulting set, if necessary, some "old" points, selected according to the rule "first in – first out," to keep the cardinality of Z_t not to exceed a given $M \ge 3$ (in our experiments we use $M = 48$). This scheme is initialized with $Z_0 = \emptyset$, $z_1 = [0; 0]$.
2. We use every run of the algorithm to obtain a set of approximate solutions to (35) associated with various values of the penalty parameter κ, as explained in **B** at the end of section 5. Precisely, when solving (35) for a given value of κ (in the sequel, we refer to it as to the *working value*, denoted κ_{w}), we also compute approximate solutions $x_\kappa(\kappa')$ to the problems with the values κ' of the penalty, for $\kappa' = \kappa\gamma$, with γ running through a given finite subset $G \ni 1$ of the positive ray. In our experiments, we used the 25-point grid $G = \{\gamma = 2^{\ell/4}\}_{\ell=-12}^{12}$.

The LO oracle for the TV norm on M_0^n utilized in COCndGM was the one described in Lemma 2; the associated flow problem (31) was solved by the commercial interior point LP solver `mosekopt` version 6 [1]. Surprisingly, in our application this "general purpose" interior point LP solver was by orders of magnitude faster than all dedicated network flow algorithms we have tried, including simplex-type network versions of `mosekopt` and `CPLEX`. With our solver, it becomes possible to replace in (31) every pair of opposite to each other arcs with a single arc, passing from the bounds $0 \le r \le \mathbf{e}$ on the flows in the arcs to the bounds $-\mathbf{e} \le r \le \mathbf{e}$.

The *termination criterion* we use relies upon the fact that in COCndGM the (nonnegative) objective decreases along the iterates: we terminate a run when the progress in terms of the objective becomes small, namely, when the condition

$$\phi_\kappa(x_{t-1}) - \phi_\kappa(x_t) \le \epsilon \max[\phi_\kappa(x_{t-1}), \delta\phi_\kappa(0)]$$

is satisfied. Here ϵ and δ are small tolerances (we used $\epsilon = 0.005$ and $\delta = 0.01$).

Organization of the experiments. In each experiment we select a "true image" $x^* \in M^n$, a kernel $\alpha(\cdot)$ and a (relative) noise intensity σ. Then we generate a related observation b, thus ending up with a particular instance of (35). This instance is solved by the outlined algorithm for working values $\kappa_{\rm w}$ of κ taken from the set $G^+ = \{\gamma = 2^{\ell/4}\}_{\ell=-\infty}^{\infty}$, with the initial working value, selected in pilot runs, of the penalty underestimating the best – resulting in the best recovery – penalty.

As explained above, a run of COCndGM, the working value of the penalty being $\kappa_{\rm w}$, yields 25 approximate solutions to (35) corresponding to κ along the grid $\kappa_{\rm w} \cdot G$. These sets are fragments of the grid G^+, with the ratio of the consecutive grid points $2^{1/4} \approx 1.19$. For every approximate solution x we compute its *combined relative error* defined as

$$\nu(x) = \left(\frac{\|\bar{x} - x^*\|_1 \|\bar{x} - x^*\|_2 \|\bar{x} - x^*\|_\infty}{\|x^*\|_1 \|x^*\|_2 \|x^*\|_\infty} \right)^{1/3};$$

here $\bar{x}$ is the easily computable shift of x by a constant image satisfying $\|\mathcal{A}\bar{x} - b\|_2 = \|P\mathcal{A}x - Pb\|_2$. From run to run, we increase the working value of the penalty by the factor $2^{1/4}$, and terminate the experiment when in four consecutive runs there was no progress in the combined relative error of the best solution found so far. Our primary goals are (a) *to quantify the performance of the COCndGM algorithm*, and (b) *to understand by which margin, in terms of $\phi_\kappa(\cdot)$, the "byproduct" approximate solutions yielded by the algorithm* (those which were obtained when solving (35) with the working value of penalty different from κ) *are worse than the "direct" approximate solution obtained for the working value κ of the penalty.*

Test instances and results. We present below the results of four experiments with two popular images; these results are fully consistent with those of other experiments we have conducted so far. The corresponding setups are presented in table 3. Table 4 summarizes the performance data. Our comments are as follows.

- In accordance to the above observations, using "large" memory (with the cardinality of Z_t allowed to be as large as 48) and processing "large" number (25) of penalty values at every step are basically costless: at an iteration, the *single* call to the LO oracle (which is a must for CndG) takes as much as 85% of the iteration time.
- The COCndGM iteration count as presented in table 4 is surprisingly low for an algorithm with sublinear $O(1/t)$ convergence, and the running time of the algorithm appears quite tolerable[11]

  Seemingly, the *instrumental* factor here is that by reasons indicated in **C**, see the end of section 5, we include into Z_t not only $z_t = [x_t; \mathrm{TV}(x_t)]$ and $\widehat{z}_t = [x[\nabla f(x_t)]; 1]$, but also $z'_t = [\nabla f(x_t); \mathrm{TV}(\nabla f(x_t))]$. To illustrate the difference, this is what happens in experiment A with the lowest (0.125) working value of penalty. With the outlined implementation, the run takes 12 iterations (111 sec), with the ratio $\phi_{1/8}(x_t)/\phi_{1/8}(x_1)$ reduced from 1 ($t = 1$) to 0.036 ($t = 12$). When z'_t is not included into Z_t, the termination criterion is not met even in 50 iterations (452 sec), the maximum iteration count we allow for a run, and in course of these 50 iterations the above ratio was reduced from 1 to 0.17, see plot e) on figure 1.

[11] For comparison: solving on the same platform problem (35) corresponding to Experiment A (256×256 image) by the state-of-the-art commercial interior point solver `mosekopt 6.0` took as much as 3,727 sec, and this – for a *single* value of the penalty (there is no clear way to get from a single run approximate solutions for a set of values of the penalty in this case).

#	Image	n	$\alpha(\cdot)$	Cond($\mathcal{A}^*\mathcal{A}$)	σ
A	`lenna`[†]	256	`fspecial('gaussian',7,1)`[§] (7×7)	$\approx 2.5e7$	0.05
B	`cameraman`[‡]	512	`fspecial('gaussian',7,1)` (7×7)	$\approx 2.5e7$	0.05
C	`lenna`	256	`fspecial('unsharp')` (3×3)	≈ 40	0.15
D	`cameraman`	512	`fspecial('unsharp')` (3×3)	≈ 40	0.40

[†]http://en.wikipedia.org/wiki/Lenna [‡]http://en.wikipedia.org/wiki/Camera_operator
[§]http://www.mathworks.com/help/images/ref/fspecial.html

Table 3: Setups of the experiments.

			Iterations per run			CPU per run, sec		CPU per iteration, sec
#	Image size	Runs	min	mean	max	mean	max	mean
A	256×256	6	4	9.00	12	83.4	148.7	8.3
B	512×512	9	4	7.89	11	212.9	318.2	25.9
C	256×256	6	17	17.17	18	189.7	214.7	10.3
D	512×512	6	16	16.00	16	615.9	768.3	36.0

Table 4: Performance of COCndGM; platform: T410 Lenovo laptop, Intel Core i7 M620 CPU@2.67GHz, 8GB RAM. Flow solver: interior point method `mosekopt 6.0` [1]

- An attractive feature of the proposed approach is the possibility to extract from a single run, the working value of the penalty being $\kappa_{\rm w}$, suboptimal solutions $x_{\kappa_{\rm w}}(\kappa)$ for a bunch of instances of (9) differing from each other by the values of the penalty κ. The related question is, of course, how good, in terms of the objective $\phi_\kappa(\cdot)$, are the "byproduct" suboptimal solutions $x_{\kappa_{\rm w}}(\kappa)$ as compared to those obtained when κ is the working value of the penalty. In our experiments, the "byproduct" solutions were pretty good, as can be seen from plots (a) – (c) on figure 1, where we see the upper and the lower envelopes of the values of ϕ_κ at the approximate solutions $x_{\kappa_{\rm w}}(\kappa)$ obtained from different working values $\kappa_{\rm w}$ of the penalty. In spite of the fact that in our experiments the ratios $\kappa/\kappa_{\rm w}$ could be as small as 1/8 and as large as 8, we see that these envelopes are pretty close to each other, and, as an additional bonus, are merely indistinguishable in a wide neighborhood of the best (resulting in the best recovery) value of the penalty (on the plots, this value is marked by asterisk).

Finally, we remark that in experiments A, B, where the mapping $\mathcal{A}$ is heavily ill-conditioned (see table 3), TV regularization yields moderate (just about 25%) improvement in the combined relative recovery error as compared to the one of the trivial recovery ("observations as they are"), in spite of the relatively low ($\sigma = 0.05$) observation noise. In contrast to this, in the experiments C, D, where $\mathcal{A}$ is well-conditioned, TV regularization reduces the error by 80% in experiment C ($\sigma = 0.15$) and by 72% in experiment D ($\sigma = 0.4$), see figure 2.

References

[1] E. D. Andersen and K. D. Andersen. The MOSEK optimization tools manual. `http://www.mosek.com/fileadmin/products/6_0/tools/doc/pdf/tools.pdf`.

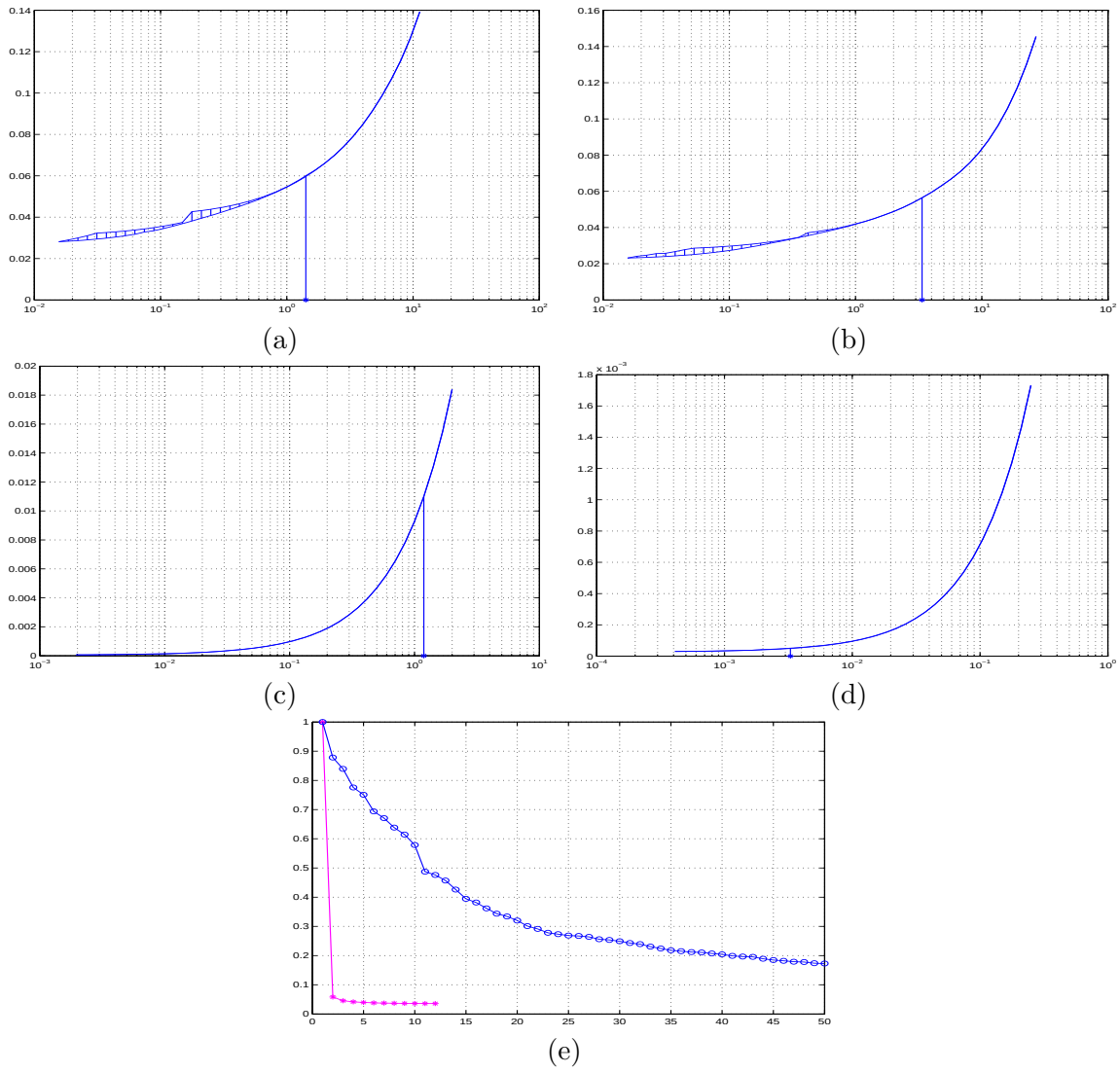
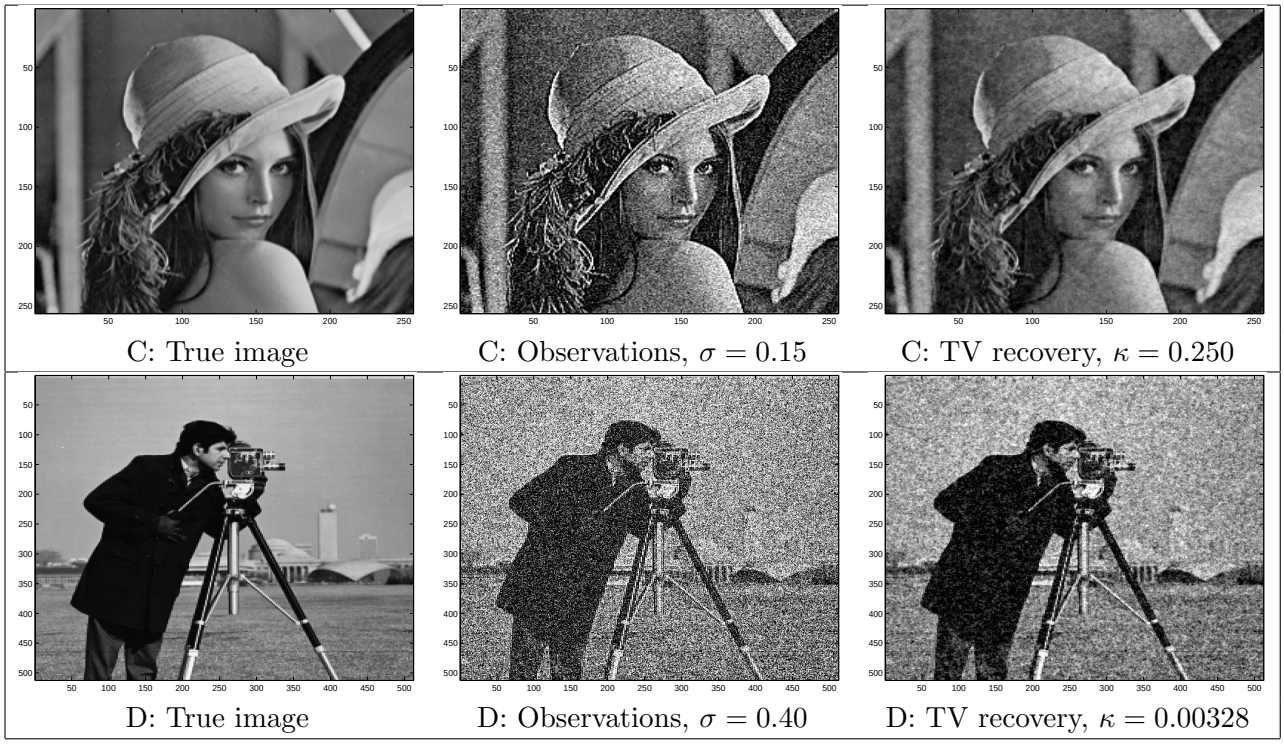


Figure 1: (a) – (d): lower and upper envelopes of $\{\phi_\kappa(x_{\kappa_w}(\kappa)) : \kappa_w \in G\}$ vs. κ, experiments A – D. Asterisks on the κ-axes: penalties resulting in the smallest combined relative recovery errors. (e): values of $\phi_{1/8}(x_t)$ vs. iteration number t with z'_t included (asterisks) and not included (circles) into Z_t.



C: True image | C: Observations, $\sigma = 0.15$ | C: TV recovery, $\kappa = 0.250$

D: True image | D: Observations, $\sigma = 0.40$ | D: TV recovery, $\kappa = 0.00328$

Figure 2: Experiments C, D

[2] F. R. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends in Machine Learning*, 2012.

[3] J.-F. Cai, E. J. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. on Optimization*, 20(4):1956–1982, 2008.

[4] E. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

[5] V. Demyanov and A. Rubinov. *Approximate Methods in Optimization Problems*. American Elsevier, 1970.

[6] M. Dudik, Z. Harchaoui, and J. Malick. Lifted coordinate descent for learning with trace-norm regularization. In *AISTATS*, 2012.

[7] J. C. Dunn and S. Harshbarger. Conditional gradient algorithms with open loop step size rules. *Journal of Mathematical Analysis and Applications*, 62(2):432–444, 1978.

[8] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.

[9] D. Goldfarb, S. Ma, and Z. Wen. Solving low-rank matrix completion problems efficiently. In *Proc. of 47th ann. Allerton conf. on Communication, control, and computing*, 2009.

[10] Z. Harchaoui, M. Douze, M. Paulin, M. Dudik, and J. Malick. Large-scale image classification with trace-norm regularization. In *CVPR*, 2012.

[11] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, 2008.

[12] D. Hearn, S. Lawphongpanich, and J. Ventura. Restricted simplicial decomposition: Computation and extensions. *Mathematical Programming Studies*, 31:99 – 118, 1987.

[13] C. Holloway. An extension of the frank-wolfe method of feasible directions. *Mathematical Programming*, 6:14 – 27, 1974.

[14] M. Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *ICML*, 2013.

[15] M. Jaggi and M. Sulovsky. A simple algorithm for nuclear norm regularized problems. In *ICML*, 2010.

[16] A. Juditsky, F. K. Karzan, and A. Nemirovski. Randomized first order algorithms with applications to ℓ_1-minimization. *Mathematical Programming*, Online First, 1 August 2012, DOI: 10.1007/s10107-012-0575-2, 2012.

[17] A. Juditsky and A. Nemirovski. First order methods for nonsmooth large-scale convex minimization, i: General purpose methods; ii: Utilizing problem's structure. In S. Sra, S. Nowozin, and S. Wright, editors, *Optimization for Machine Learning*, pages 121–184. The MIT Press, 2012.

[18] G. Lan. An optimal method for stochastic composite optimization. *Math. Program.*, 2012.

[19] C. Lemaréchal, A. Nemirovskii, and Y. Nesterov. New variants of bundle methods. *Mathematical Programramming*, 1995.

[20] S. Ma, D. Goldfarb, and L. Chen. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 128:321 – 353, 2011.

[21] A. S. Nemirovski and D. B. Yudin. *Problem complexity and method efficiency in optimization*. Wiley-Interscience, 1983.

[22] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, 2003.

[23] Y. Nesterov. Gradient methods for minimizing composite objective function. Technical Report 76, CORE Discussion Paper, 2007.

[24] Y. Nesterov and A. Nemirovski. On first order algorithms for ℓ_1/nuclear norm minimization. *to appear in Acta Numerica*, 22, 2013.

[25] B. Pshenichnyj and Y. Danilin. *Numerical Methods in Extremal Problems*. Mir, 1978.

[26] B. Recht, M. Fazel, and P. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

[27] B. Recht and C. Ré. Parallel stochastic gradient algorithms for large-scale matrix completion. Technical Report pages.cs.wisc.edu/∼brecht/papers/11.Rec.Re.IPGM.pdf, Preprint, Computer Sciences Department, University of Wisconsin-Madison, 2011.

[28] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60, 1992.

[29] S. Shalev-Shwartz, A. Gonen, and O. Shamir. Large-scale convex minimization with a low-rank constraint. In *ICML*, 2011.

[30] S. Sra, S. Nowozin, and S. J. Wright. *Optimization for Machine Learning*. MIT Press, 2010.

[31] N. Srebro and A. Shraibman. Rank, trace-norm and max-norm. In *COLT*, 2005.

[32] J. A. Ventura and D. W. Hearn. Restricted simplicial decomposition for convex constrained problems. *Mathematical Programming*, 59:71 – 85, 1993.

[33] J. Yang and X. Yan. Linearized augmented lagrangian and alternating direction methods for nuclear norm minimization. Technical Report 2010/02/2534, www.optimization-online.org, 2011. `http://www.optimization-online.org/DB_FILE/2010/02/2534.pdf`.

[34] M. Zibulevski and G. Narkiss. Sequential subspace optimization method for large-scale unconstrained problems. Technical Report Tech. Report CCIT No 559, Faculty of Electrical engineering, Technion, 2005.

8 Appendix

8.1 Proof of Theorem 1

Define

$$\epsilon_t = f(x_t) - f_*, \quad \Delta_t = \max_{x\in X}\langle f'(x_t), x_t - x\rangle = \langle f'(x_t), x_t - x_t^+\rangle$$

where $x_t^+ = x_X[f'(x_t)]$. Denoting by x_* an optimal solution to (13) and invoking the definition of x_t^+ and convexity of f, we have

$$\langle f'(x_t), x_t^+ - x_t\rangle \le \langle f'(x_t), x_* - x_t\rangle \le f_* - f(x_t). \tag{37}$$

Observing that for a generic GC algorithm we have $f(x_{t+1}) \le f(x_t + \gamma_t(x_t^+ - x_t))$ and invoking (12), we have

$$f(x_{t+1}) \le f(x_t) + \gamma_t\langle f'(x_t), x_t^+ - x_t\rangle + \frac{L}{2}\gamma_t^2\|x_t^+ - x_t\|_X^2 \le f(x_t) - \gamma_t(f(x_t) - f_*) + \tfrac{1}{2}L\gamma_t^2, \tag{38}$$

where the concluding $\le$ is due to (37). It follows that $\epsilon_{t+1} \le (1-\gamma_t)\epsilon_t + \frac{1}{2}L\gamma_t^2$, whence

$$\begin{aligned}\epsilon_{t+1} &\le \epsilon_1\prod_{i=1}^t(1-\gamma_i) + \tfrac{1}{2}L\sum_{i=1}^t\gamma_i^2\prod_{k=i+1}^t(1-\gamma_k)\\ &= 2L\sum_{i=1}^t(i+1)^{-2}\prod_{k=i+1}^t(1-\frac{2}{k+1}),\end{aligned}$$

where, by convention, $\prod_{k=t+1}^t = 1$. Noting that $\prod_{k=i+1}^t(1-\frac{2}{k+1}) = \prod_{k=i+1}^t\frac{k-1}{k+1} = \frac{i(i+1)}{t(t+1)}$, $\quad i = 1, ..., t$, we get

$$\epsilon_{t+1} \le 2L\sum_{i=1}^t\frac{i(i+1)}{(i+1)^2t(t+1)} \le \frac{2Lt}{(t+1)^2} \le 2L(t+2)^{-1}, \tag{39}$$

what is (19).

To prove (20), observe that setting $\bar\Delta_t = \min_{1\le k\le t}\Delta_k$, and invoking (17), (18) we clearly have

$$\begin{aligned}f(\bar x_t) - f_t^* &= \min_{1\le k\le t}[f(\bar x_t) - f_{*,k}] = \min_{1\le k\le t}[f(\bar x_t) - f(x_k) + \Delta_k]\\ &\le \min_{1\le k\le t}\Delta_k = \bar\Delta_t\end{aligned}$$

(we have used the fact that $f(\bar x_t) \le f(x_k)$, $k \le t$, by definition of $\bar x_t$). We see that in order to prove (20), it suffices to prove that

$$\bar\Delta_t \le \frac{4.5L}{t-2},\ t = 5, 6, ... \tag{40}$$

To verify (40), note that by the first inequality in (38)

$$\gamma_k\Delta_k \equiv \gamma_k\langle f'(x_k), x_k - x_k^+\rangle \le \epsilon_k - \epsilon_{k+1} + \tfrac{1}{2}L\gamma_k^2. \tag{41}$$

Assuming $t > 2$ and summing up these inequalities over k varying from $t_0 = \lfloor t/2\rfloor$ to t (here $\lfloor a\rfloor$ stands for the largest integer strictly smaller than a), we obtain

$$\sum_{k=t_0}^t\gamma_k\Delta_k \le \epsilon_{t_0} + \tfrac{1}{2}L\sum_{k=t_0}^t\gamma_k^2,$$

and therefore

$$\bar{\Delta}_t \sum_{k=t_0}^{t} \gamma_k \le \epsilon_{t_0} + \tfrac{1}{2} L \sum_{k=t_0}^{t} \gamma_k^2. \tag{42}$$

Observe that $\sum_{k=t_0}^{t} \gamma_k = 2\sum_{k=t_0}^{t}(k+1)^{-1} \ge 2[\ln(t+1) - \ln(t_0+1)] \ge 2\ln(2)$ and $\sum_{k=t_0}^{t} \gamma_k^2 = 4\sum_{k=t_0}^{t}(k+1)^{-2} \le 4[t_0^{-1} - t^{-1}] \le \frac{4(t+2)}{t(t-2)}$. Assuming $t > 4$ (so that $t_0 \ge 2$) and substituting into (42) the bound (19) for ϵ_{t_0} we obtain

$$\bar{\Delta}_t \le \frac{2L}{t} + \frac{L(t+2)}{t(t-2)\ln(2)} \le 4.5L(t-2)^{-1},$$

as required in (40). □

8.2 Proof of Theorem 2

The proof, up to minor modifications, goes back to [19], see also [22, 16]; we provide it here to make the paper self-contained. W.l.o.g. we can assume that we are in the nontrivial case (see description of the algorithm).

1^0. As it was explained when describing the method, whenever stage s takes place, we have $[0 <]\rho_1 \le \rho_s \le \rho_*$, and $\rho_{s-1} < \rho_s$, provided $s > 1$. Therefore by the termination rule, the output $\bar{\rho}$, $\bar{x}$ of the algorithm, if any, satisfies $\bar{\rho} \le \rho_*$, $f(\bar{x}) \le \epsilon$. Thus, (i) holds true, provided that the algorithm does terminate. Thus, all we need is to verify (ii) and (iii).

2^0. Let us prove (ii). Let $s \ge 1$ be such that stage s takes place. Setting $X = K[\rho_s]$, observe that $X - X \subset \{x \in E : \|x\| \le 2\rho_s\}$, whence $\|\cdot\| \le 2\rho_s\|\cdot\|_X$, and therefore the relation (4) implies the validity of (12) with $L = 4\rho_s^2 L_f$. Now, if stage s does not terminate in course of some number t steps, then, in the notation from the description of the algorithm, $f(\bar{x}_t) > \epsilon$ and $f_*^t < \frac{3}{4} f(\bar{x}_t)$, whence $f(\bar{x}_t) - f_*^t > \epsilon/4$. By Theorem 1.ii, the latter is possible only when $4.5L/(t-2) > \epsilon/4$. Thus, $t \le \max\left[5, 2 + \frac{72\rho_s^2 L_f}{\epsilon}\right]$. Taking into account that $\rho_s \le \rho_*$, (ii) follows.

3^0. Let us prove (iii). This statement is trivially true when the number of stages is 1. Assuming that it is not the case, let $S \ge 1$ be such that the stage $S+1$ takes place. For every $s = 1, ..., S$, let t_s be the last step of stage s, and let u_s, $\ell^s(\cdot)$ be what in the notation from the description of stage s was denoted $f(\bar{x}_{t_s})$ and $\ell^{t_s}(\rho)$. Thus, $u_s > \epsilon$ is an upper bound on $\mathrm{Opt}(\rho_s)$, $\ell_s := \ell^s(\rho_s)$ is a lower bound on $\mathrm{Opt}(\rho_s)$ satisfying $\ell_s \ge 3u_s/4$, and $\ell^s(\cdot)$ is a piecewise linear convex in ρ lower bound on $\mathrm{Opt}(\rho)$, $\rho \ge 0$, and $\rho_{s+1} > \rho_s$ is the smallest positive root of $\ell^s(\cdot)$. Let also $-g_s$ be a subgradient of $\ell^s(\cdot)$ at ρ_s. Note that $g_s > 0$ due to $\rho_{s+1} > \rho_s$ combined with $\ell^s(\rho_s) > 0$, $\ell^s(\rho_{s+1}) = 0$, and by the same reasons combined with convexity of $\ell^s(\cdot)$ we have

$$\rho_{s+1} - \rho_s \ge \ell_s / g_s, \tag{43}$$

and, as we have seen,

$$1 \le s \le S \Rightarrow \left\{ \begin{array}{ll} (a) & u_s > \epsilon, \\ (b) & u_s \ge \mathrm{Opt}(\rho_s) \ge \ell_s \ge \frac{3}{4} u_s, \\ (c) & \ell_s - g_s(\rho - \rho_s) \le \mathrm{Opt}(\rho),\ \rho \ge 0. \end{array} \right. . \tag{44}$$

Assuming $1 < s \le S$ and applying (43), we get $\rho_s - \rho_{s-1} \ge \frac{3}{4}u_{s-1}/g_{s-1}$, whence, invoking (44),

$$u_{s-1} \ge \mathrm{Opt}(\rho_{s-1}) \ge \ell_s + g_s[\rho_{s-1} - \rho_s] \ge \frac{3}{4}u_s + \frac{3}{4}u_{s-1}\frac{g_s}{g_{s-1}}.$$

The resulting inequality implies that $\frac{u_s}{u_{s-1}} + \frac{g_s}{g_{s-1}} \le \frac{4}{3}$, whence $\frac{u_s g_s}{u_{s-1}g_{s-1}} \le (1/4)(4/3)^2 = 4/9$. It follows that

$$\sqrt{u_s g_s} \le (2/3)^{s-1}\sqrt{u_1 g_1},\ 1 \le s \le S. \tag{45}$$

Now, since the first iterate of the first stage is 0, we have $u_1 \le f(0)$, while (44) applied with $s = 1$ implies that $f(0) = \mathrm{Opt}(0) \ge \ell_1 + \rho_1 g_1 \ge \rho_1 g_1$, whence $u_1 g_1 \le f(0)/\rho_1 = d$. Further, by (43) $g_s \ge \ell_s/(\rho_{s+1} - \rho_s) \ge q\ell_s/\rho_* \ge \frac{3}{4}u_s/\rho_*$, where the concluding inequality is given by (44). We see that $u_s g_s \ge \frac{3}{4}u_s^2/\rho_* \ge \frac{3}{4}\epsilon^2/\rho_*$. This lower bound on $u_s g_s$ combines with the bound $u_1 g_1 \le d$ and with (45) to imply that

$$\epsilon \le \sqrt{4/3}(2/3)^{s-1}\sqrt{d\rho_*},\ 1 \le s \le S.$$

Finally observe that by the definition of ρ_* and due to the fact that $\|x[f'(0)]\| = 1$ in the nontrivial case, we have

$$0 \le f(\rho_* x[f'(0)]) \le f(0) + \rho_*\langle f'(0), x[f'(0)]\rangle + \frac{1}{2}L_f\rho_*^2 = f(0) - \rho_* d + \tfrac{1}{2}L_f\rho_*^2$$

(we have used (4) and the definition of d), whence $\rho_* d \le f(0) + \frac{1}{2}L_f\rho_*^2$ and therefore

$$\epsilon \le \sqrt{3/4}(2/3)^{s-1}\sqrt{f(0) + \tfrac{1}{2}L_f\rho_*^2},\ 1 \le s \le S.$$

Since this relation holds true for every $S \ge 1$ such that the stage $S+1$ takes place, (iii) follows. □

8.3 Proof of Theorem 3

By definition of z_t we have $z_t \in K^+$ for all t and $F(0) = F(z_1) \ge F(z_2) \ge ...$, whence $r_t \le D_*$ for all t by Assumption A. Besides this, $r_* \le D_*$ as well. Let now $\epsilon_t = F(z_t) - F_*$, $z_t = [x_t; r_t]$, and let $z_t^+ = [x_t^+, r_t^+]$ be a minimizer, as given by Lemma 1, of the linear form $\langle F'(z_t), z\rangle$ of $z \in E^+$ over the set $K^+[r_*] = \{[x; r] : x \in K, \|x\| \le r \le r_*\}$. Recalling that $F'(z_t) = [f'(x_t); \kappa]$ and that $r_t \le D_* \le D^+$, Lemma 1 implies that $z_t^+ \in \Delta(z_t)$. By definition of z_t^+ and convexity of F we have

$$\begin{array}{rcl}\langle [f'(x_t);\kappa], z_t - z_t^+\rangle & = & \langle f'(x_t), x_t - x_t^+\rangle + \kappa(r_t - r_t^+)\\ & \ge & \langle f'(x_t), x_t - x_*\rangle + \kappa(r_t - r_*)\\ & = & \langle F'(z_t),\ z_t - z_*\rangle \ge F(z_t) - F(z_*) = \epsilon_t.\end{array}$$

Invoking (12), it follows that for $0 \le s \le 1$ one has

$$\begin{array}{rcl}F(z_t + s(z_t^+ - z_t)) & \le & F(z_i) + s\langle [f'(x_t);\kappa], z_t^+ - z_t\rangle + \dfrac{L_f s^2}{2}\|x(z_t^+) - x(z_t)\|^2\\ & \le & F(z_t) - s\epsilon_t + \frac{1}{2}L_f s^2(r_t + D_*)^2\end{array}$$

using that $\|x(z_t^+)\| \le r_t^+$ and $\|x(z^t)\| \le r_t$ due to $z_t^+, z_t \in K^+$, and that $r_t^+ \le r_* \le D_*$. By (24) we have

$$F(z_{t+1}) \le \min_{0\le s\le 1} F(z_t + s(z_t^+ - z_t)) \le F(z_t) + \min_{0\le s\le 1}\left\{-s\epsilon_t + \tfrac{1}{2}L_f s^2(r_t + D_*)^2\right\},$$

and we arrive at the recurrence

$$\epsilon_{t+1} \le \epsilon_t - \left\{ \begin{array}{ll} \frac{\epsilon_t^2}{2L_f(r_t+D_*)^2}, & \epsilon_t \le L_f(r_t+D_*)^2 \\ \epsilon_t - \frac{1}{2}L_f(r_t+D_*)^2, & \epsilon_t > L_f(r_t+D_*)^2 \end{array} \right. , t = 1, 2, ... \tag{46}$$

When $t = 1$, this recurrence, in view of $z_1 = 0$, implies that $\epsilon_2 \le \frac{1}{2}L_f D_*^2$. Let us show by induction in $t \ge 2$ that

$$\epsilon_t \le \bar{\epsilon}_t := \frac{8L_f D_*^2}{t+14}, \, t = 2, 3, ... \tag{47}$$

thus completing the proof. We have already seen that (47) is valid for $t = 2$. Assuming that (47) holds true for $t = k \ge 2$, we have $\epsilon_k \le \frac{1}{2}L_f D_*^2$ and therefore $\epsilon_{k+1} \le \epsilon_k - \frac{1}{8L_f D_*^2}\epsilon_k^2$ by (46) combined with $0 \le r_k \le D_*$. Now, the function $s - \frac{1}{8L_f D_*^2}s^2$ is nondecreasing on the segment $0 \le s \le 4L_f D_*^2$ which contains $\bar{\epsilon}_k$ and $\epsilon_k \le \bar{\epsilon}_k$, whence

$$\begin{array}{rcl} \epsilon_{k+1} & \le & \epsilon_k - \frac{1}{8L_f D_*^2}\epsilon_k^2 \le \bar{\epsilon}_k - \frac{1}{8L_f D_*^2}\bar{\epsilon}_k^2 = \left[\frac{8L_f D_*^2}{k+14}\right] - \frac{1}{8L_f D_*^2}\left[\frac{8L_f D_*^2}{k+14}\right]^2 \\ & = & \frac{8L_f D_*^2(k+13)}{(k+14)^2} \le \frac{8L_f D_*^2}{(k+1)+14}, \end{array}$$

so that (47) holds true for $t = k + 1$. □

8.4 Proofs for Section 6

Proof of Lemma 2. As we have already explained, (31) is solvable, so that z is well defined. Denoting by (s^*, r^*) an optimal solution to (31) produced, along with z, by our solver, note that the characteristic property of z is the relation

$$(s^*, r^*) \in \operatorname*{Argmax}_{s,r} \{s + \langle z, Pr - s\eta \rangle : 0 \le r \le \mathbf{e}\}.$$

Since the column sums in P are zeros and η is with zero sum of entries, the above characteristic property of z is preserved when passing from z to $\bar{z}$, so that we may assume from the very beginning that $z = \bar{z}$ is a zero mean image. Now, $P = [Q, -Q]$, where Q is the incidence matrix of the network obtained from G by eliminating backward arcs. Representing a flow r as $[r_f; r_b]$, where the blocks are comprised, respectively, of flows in the forward and backward arcs, and passing from r to $\rho = r_f - r_b$, our characteristic property of z clearly implies the relation

$$(s^*, \ \rho^* := r_f^* - r_b^*) \in \operatorname*{Argmax}_{s,\rho} \{s + \langle z, Q\rho - s\eta \rangle : \|\rho\|_\infty \le 1\}.$$

It follows that

$$\begin{array}{ll} (a) & \langle z, \eta \rangle = 1, \\ (b) & \|\rho^*\|_\infty \le 1, \\ (c) & (Q^* z)_\gamma = \left\{ \begin{array}{ll} \le 0, & \rho_\gamma^* = -1, \\ = 0, & \rho_\gamma^* \in (-1, 1), \\ \ge 0, & \rho_\gamma^* = 1, \end{array} \right. \quad \text{for all forward arcs } \gamma, \\ (d) & Q\rho^* = s^*\eta. \end{array} \tag{48}$$

(48.d) and (48.a) imply that $\langle Q^*z, \rho^*\rangle = s^*$, while (48.$c$) says that $\langle Q^*z, \rho^*\rangle = \|Q^*z\|_1$, and $s^* = \|Q^*z\|_1$. By (48.a) $z \neq 0$, and thus z is a nonzero image with zero mean; recalling what Q is, the first $n(n-1)$ entries in Q^*z form $\nabla_i z$, and the last $n(n-1)$ entries form $\nabla_j z$, so that $\|Q^*z\|_1 = \mathrm{TV}(z)$. The gradient field of a nonzero image with zero mean cannot be identically zero, whence $\mathrm{TV}(z) = \|Q^*z\|_1 = s^* > 0$. Thus $x[\eta] = -z/\mathrm{TV}(z) = -z/s^*$ is well defined and $\mathrm{TV}(x[\eta]) = 1$, while by (48.a) we have $\langle x[\eta], \eta\rangle = -1/s^*$. Finally, let $x \in \mathcal{TV}$, implying that Q^*x is the concatenation of $\nabla_i x$ and $\nabla_j x$ and thus $\|Q^*x\|_1 = \mathrm{TV}(x) \leq 1$. Invoking (48.$b, d$), we get $-1 \leq \langle Q^*x, \rho^*\rangle = \langle x, Q\rho^*\rangle = s^*\langle x, \eta\rangle$, whence $\langle x, \eta\rangle \geq -1/s^* = \langle x[\eta], \eta\rangle$, meaning that $x[\eta] \in \mathcal{TV}$ is a minimizer of $\langle \eta, x\rangle$ over $x \in \mathcal{TV}$. □

Proof of Proposition 1. In the sequel, for a real-valued function x defined on a finite set (e.g., for an image), $\|x\|_p$ stands for the L_p norm of the function corresponding to the counting measure on the set (the mass of every point from the set is 1). Let us fix n and $x \in M_0^n$ with $\mathrm{TV}(x) \leq 1$; we want to prove that

$$\|x\|_2 \leq \mathcal{C}\sqrt{\ln(n)} \tag{49}$$

with appropriately selected *absolute constant* $\mathcal{C}$.

1^0. Let $\oplus$ stand for addition, and $\ominus$ – for substraction of integers modulo n; $p \oplus q = (p + q) \bmod n \in \{0, 1, ..., n-1\}$ and similarly for $p \ominus q$. Along with discrete partial derivatives $\nabla_i x$, $\nabla_j x$, let us define their periodic versions $\widehat{\nabla}_i x$, $\widehat{\nabla}_j x$:

$$\widehat{\nabla}_i x(i,j) = x(i \oplus 1, j) - x(i,j) : \Gamma_{n,n} \to \mathbf{R}, \ \widehat{\nabla}_j x(i,j) = x(i, j \oplus 1) - x(i,j) : \Gamma_{n,n} \to \mathbf{R},$$

same as periodic Laplacian $\widehat{\Delta} x$:

$$\widehat{\Delta} x = x(i,j) - \frac{1}{4}\left[x(i \ominus 1, j) + x(i \oplus 1, j) + x(i, j \ominus 1) + x(i, j \oplus 1)\right] : \Gamma_{n,n} \to \mathbf{R}.$$

For every j, $0 \leq j < n$, we have $\sum_{i=0}^{n-1} \widehat{\nabla}_i x(i,j) = 0$ and $\nabla_i x(i,j) = \widehat{\nabla}_i x(i,j)$ for $0 \leq i < n-1$, whence $\sum_{i=0}^{n-1} |\widehat{\nabla}_i(x)| \leq 2\sum_{i=0}^{n-1} |\nabla_i x(i,j)|$ for every j, and thus $\|\widehat{\nabla}_i x\|_1 \leq 2\|\nabla_i x\|_1$. Similarly, $\|\widehat{\nabla}_j x\|_1 \leq 2\|\nabla_j x\|_1$, and we conclude that

$$\|\widehat{\nabla}_i x\|_1 + \|\widehat{\nabla}_j x\|_1 \leq 2. \tag{50}$$

2^0. Now observe that for $0 \leq i, j < n$ we have

$$\begin{array}{rcl} x(i,j) & = & x(i \ominus 1, j) + \widehat{\nabla}_i x(i \ominus 1, j) \\ x(i,j) & = & x(i \oplus 1, j) - \widehat{\nabla}_i x(i,j) \\ x(i,j) & = & x(i, j \ominus 1) + \widehat{\nabla}_j x(i, j \ominus 1) \\ x(i,j) & = & x(i, j \oplus 1) - \widehat{\nabla}_j x(i,j) \end{array}$$

whence

$$\widehat{\Delta} x(i,j) = \frac{1}{4}\left[\widehat{\nabla}_i x(i \ominus 1, j) - \widehat{\nabla}_i x(i,j) + \widehat{\nabla}_j x(i, j \ominus 1) - \widehat{\nabla}_j x(i,j)\right] \tag{51}$$

Now consider the following linear mapping from $M^n \times M^n$ into M^n:

$$B[g,h](i,j) = \frac{1}{4}\left[g(i \ominus 1, j) - g(i,j) + h(i, j \ominus 1) - h(i,j)\right], \ [i;j] \in \Gamma_{n,n}. \tag{52}$$

From this definition and (51) it follows that

$$\widehat{\Delta}x = B[\widehat{\nabla}_i x, \widehat{\nabla}_j x]. \tag{53}$$

3^0. Observe that $B[g,h]$ always is an image with zero mean. Further, passing from images $u \in M^n$ to their 2D Discrete Fourier Transforms DFT$[u]$:

$$\mathrm{DFT}[u](p,q) = \sum_{0\le r,s<n} u(r,s)\exp\{-2\pi\imath(pr+qs)/n\},\ [p;q]\in\Gamma_{n,n},$$

we immediately see that every image u with zero mean is the periodic Laplacian of another, uniquely, defined, image $X[u]$ with zero mean, with $X[u]$ given by

$$\begin{gathered}\mathrm{DFT}[X[u]](p,q) = Y[u](p,q) := \begin{cases} 0, & p=q=0 \\ \frac{\mathrm{DFT}[u](p,q)}{D(p,q)}, & 0\ne[p;q]\in\Gamma_{n,n}\end{cases},\ [p;q]\in\Gamma_{n,n},\\ D(p,q) = 1-\tfrac{1}{2}[\cos(2\pi p/n)+\cos(2\pi q/n)],\ [p;q]\in\Gamma_{n,n}.\end{gathered} \tag{54}$$

In particular, invoking (53), we get

$$\mathrm{DFT}[x] = Y[B[\widehat{\nabla}_i x, \widehat{\nabla}_j x]].$$

By Parseval identity, $\|\mathrm{DFT}[x]\|_2 = n\|x\|_2$, whence

$$\|x\|_2 = n^{-1}\|Y[B[\widehat{\nabla}_i x, \widehat{\nabla}_j x]]\|_2.$$

Combining this observation with (50), we see that in order to prove (49), it suffices to check that

(!) *Whenever $g,h\in M^n$ are such that*

$$(g,h)\in G := \{(g,h)\in M^n\times M^n : \|g\|_1+\|h\|_1\le 2\},$$

we have

$$\|Y[B[g,h]]\|_2 \le n\mathcal{C}\sqrt{\ln(n)}. \tag{55}$$

4^0. A good news about (!) is that since $Y[B[g,h]]$ is linear in (g,h), in order to justify (!), it suffices to prove that (55) holds true for the extreme point of G, i.e., (a) for pairs where $h\equiv 0$ and g is an image which is equal to 2 at some point of $\Gamma_{n,n}$ and vanishes outside of this point, and (b) for pairs where $g\equiv 0$ and h is an image which is equal to 2 at some point of $\Gamma_{n,n}$ and vanishes outside of this point. Task (b) clearly reduces to task (a) by swapping the coordinates i,j of points from $\Gamma_{n,n}$, so that we may focus solely on task (a). Thus, assume that g is a cyclic shift of the image 2δ:

$$g(i,j)\equiv 2\delta(i\ominus r, j\ominus s),\ \delta(i,j) = \begin{cases} 1, & [i;j]=[0;0] \\ 0, & [i;j]\ne[0;0]\end{cases},\ [i;j]\in\Gamma_{n,n}.$$

From (52) it follows that then $B[g,0]$ is a cyclic shift of $B[2\delta,0]$, whence $|\mathrm{DFT}[B[g,0]](p,q)| = |\mathrm{DFT}[B[2\delta,0]](p,q)|$ for all $[p;q]\in\Gamma_{n,n}$, which, by (54), implies that $|Y[B[g,0]](p,q)| = |Y[B[2\delta,0]](p,q)|$ for all $[p;q]\in\Gamma_{n,n}$. The bottom line is that *all we need is to verify that* (55) *holds true for* $g=2\delta, h=0$, or, which is the same, that with

$$y(p,q) = \frac{(1-\exp\{2\pi\imath p/n\})}{2[1-\frac{1}{2}[\cos(2\pi p/n)+\cos(2\pi q/n)]]} \tag{56}$$

where the right hand side by definition is 0 at $p = q = 0$, it holds

$$C_n := \sum_{p,q=0}^{n-1} |y(p,q)|^2 \leq n^2\mathcal{C}^2 \ln(n).$$

Now, (56) makes sense for all $[p;q] \in \mathbf{Z}^2$ (provided that we define the right hand side as zero at all points of $\mathbf{Z}^2$ where the denominator in (56) vanishes, that is, at all point where p, q are integer multiples of n) and defines y as a double-periodic, with periods n in p and in q, function of $[p;q]$. Therefore, setting $m = \mathrm{Floor}(n/2) \geq 1$ and $W = \{[p;q] \in \mathbf{Z}^2 : -m \leq p, q < n - m\}$, we have

$$C_n = \sum_{0\neq[p;q]\in W} |y(p,q)|^2 = \sum_{[p;q]\in W} \frac{|1 - \exp\{2\pi\imath p/n\}|^2}{4|1 - \frac{1}{2}[\cos(2\pi p/n) + \cos(2\pi q/n)]|^2}.$$

Setting $\rho(p,q) = \sqrt{p^2+q^2}$, observe that when $0 \neq [p;q] \in W$, we have $|1 - \exp\{2\pi\imath p/n\}| \leq C_1 n^{-1}\rho(p,q)$ and $2[1-\frac{1}{2}[\cos(2\pi p/n)+\cos(2\pi q/n)]] \geq C_2 n^{-2}\rho^2(p,q)$ with positive absolute constants C_1, C_2, whence

$$C_n \leq (C_1/C_2)^2 \sum_{0\neq[p;q]\in W} n^2\rho^{-2}(p,q).$$

With appropriately selected absolute constant C_3 we have

$$\sum_{0\neq[p;q]\in W} \rho^{-2}(p,q) \leq C_3 \int_1^n r^{-2} r dr = C_3 \ln(n).$$

Thus, $C_n \leq (C_1/C_2)^2 C_3 n^2 \ln(n)$, meaning that (55), and thus (49), holds true with $\mathcal{C} = \sqrt{C_3}C_1/C_2$. □