To appear in Mathematical Programming

Noname manuscript No.
(will be inserted by the editor)

Optimized first-order methods for smooth convex minimization

Donghwan Kim · Jeffrey A. Fessler

Date of current version: September 9, 2015

Abstract We introduce new optimized first-order methods for smooth unconstrained convex minimization. Drori and Teboulle [5] recently described a numerical method for computing the N-iteration optimal step coefficients in a class of first-order algorithms that includes gradient methods, heavy-ball methods [15], and Nesterov's fast gradient methods [10,12]. However, the numerical method in [5] is computationally expensive for large N, and the corresponding numerically optimized first-order algorithm in [5] requires impractical memory and computation for large-scale optimization problems. In this paper, we propose optimized first-order algorithms that achieve a convergence bound that is two times smaller than for Nesterov's fast gradient methods; our bound is found analytically and refines the numerical bound in [5]. Furthermore, the proposed optimized first-order methods have efficient forms that are remarkably similar to Nesterov's fast gradient methods.

Keywords First-order algorithms · Convergence bound · Smooth convex minimization · Fast gradient methods

1 Introduction

First-order algorithms are used widely to solve large-scale optimization problems in various fields such as signal and image processing, machine learning, communications and many other areas. The computational cost per iteration of first-order

This research was supported in part by NIH grant R01-HL-098686 and U01-EB-018753.

Donghwan Kim · Jeffrey A. Fessler
Dept. of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA
E-mail: kimdongh@umich.edu, fessler@umich.edu

algorithms is mildly dependent on the dimension of the problem, yielding computational efficiency. Particularly, Nesterov's fast gradient methods [10,12] have been celebrated in various applications for their fast convergence rates and efficient implementation. This paper proposes first-order algorithms (OGM1 and OGM2 in Section 7) that achieve a worst-case convergence bound that is twice as small as Nesterov's fast gradient methods for smooth unconstrained convex minimization yet have remarkably similar efficient implementations.

We consider finding a minimizer over $\mathbb{R}^d$ of a cost function f belonging to a set $\mathcal{F}_L(\mathbb{R}^d)$ of smooth convex functions with L-Lipschitz continuous gradient. The class of first-order (FO) algorithms of interest generates a sequence of points $\{\boldsymbol{x}_i \in \mathbb{R}^d \ : \ i = 0, \cdots, N\}$ using the following scheme:

```
Algorithm Class FO
    Input: $f \in \mathcal{F}_L(\mathbb{R}^d)$, $\boldsymbol{x}_0 \in \mathbb{R}^d$.
    For $i = 0, \cdots, N-1$
        $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \frac{1}{L}\sum_{k=0}^{i} h_{i+1,k} f'(\boldsymbol{x}_k).$        (1.1)
```

The update step at the ith iterate $\boldsymbol{x}_i$ uses a linear combination of previous and current gradients $\{f'(\boldsymbol{x}_0), \cdots, f'(\boldsymbol{x}_i)\}$. The coefficients $\{h_{i,k}\}_{0 \le k < i \le N}$ determine the step size and are selected prior to iterating (non-adaptive). Designing these coefficients appropriately is the key to establishing fast convergence. The algorithm class FO includes gradient methods, heavy-ball methods [15], Nesterov's fast gradient methods [10,12], and our proposed optimized first-order methods.

Evaluating the convergence bound of such first-order algorithms is essential. Recently, Drori and Teboulle (hereafter "DT") [5] considered the Performance Estimation Problem (PEP) approach to bounding the decrease of a cost function f. For given coefficients $\boldsymbol{h} = \{h_{i,k}\}_{0 \le k < i \le N}$, a given number of iterations $N \ge 1$ and a given upper bound $R > 0$ on the distance between an initial point $\boldsymbol{x}_0$ and an optimal point $\boldsymbol{x}_* \in X_*(f) \triangleq \arg\min_{\boldsymbol{x} \in \mathbb{R}^d} f(\boldsymbol{x})$, the worst-case performance bound of a first-order method over all smooth convex functions $f \in \mathcal{F}_L(\mathbb{R}^d)$ is the solution of the following constrained optimization problem:[1] [5]:

$$\mathcal{B}_{\text{P}}(\boldsymbol{h}, N, d, L, R) \triangleq \max_{f \in \mathcal{F}_L(\mathbb{R}^d)} \max_{\substack{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_N \in \mathbb{R}^d, \\ \boldsymbol{x}_* \in X_*(f)}} f(\boldsymbol{x}_N) - f(\boldsymbol{x}_*) \tag{P}$$

$$\text{s.t. } \boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \frac{1}{L}\sum_{k=0}^{i} h_{i+1,k} f'(\boldsymbol{x}_k), \quad i = 0, \cdots, N-1,$$

$$||\boldsymbol{x}_0 - \boldsymbol{x}_*|| \le R.$$

As reviewed in Section 4, DT [5] used relaxations to simplify the intractable problem (P) to a solvable form.

[1] The problem $\mathcal{B}_{\text{P}}(\boldsymbol{h}, N, d, L, R)$ was shown to be independent of d in [17]; thus this paper's results are independent of d.

Nesterov's fast gradient methods [10,12] achieve the optimal rate of decrease $O\left(\frac{1}{N^2}\right)$ for minimizing a smooth convex function f [11]. Seeking first-order algorithms that converge faster (in terms of the constant factor) than Nesterov's fast gradient methods, DT [5] proposed using a (relaxed) PEP approach to optimize the choice of $\boldsymbol{h}$ in class FO by minimizing a (relaxed) worst-case bound at the Nth iteration with respect to $\boldsymbol{h}$. In [5], the optimized $\boldsymbol{h}$ factors were computed numerically, and were found to yield faster convergence than Nesterov's methods. However, numerical optimization of $\boldsymbol{h}$ in [5] becomes expensive for large N. In addition, the general class FO requires $O(N^2d)$ arithmetic operations for N iterations and $O(Nd)$ memory for storing all gradients $\{f'(\boldsymbol{x}_i) \in \mathbb{R}^d \ : \ i = 0, \cdots, N-1\}$, which is impractical for large-scale problems.

This paper proposes optimized first-order algorithms that have a worst-case convergence bound that is twice as small as that of Nesterov's fast gradient methods, inspired by [5]. We develop remarkably efficient formulations of the optimized first-order algorithms that resemble those of Nesterov's fast gradient methods, requiring $O(Nd)$ arithmetic operations and $O(d)$ memory.

Section 2 reviews the smooth convex minimization problem and introduces the approach to optimizing $\boldsymbol{h}$ used here and in [5]. Section 3 illustrates Nesterov's fast gradient methods that are in class FO. Section 4 reviews DT's (relaxed) PEP approach and Section 5 uses it to derive a new convergence bound for the *secondary* variables in Nesterov's fast gradient methods. Section 6 reviews DT's analysis on numerically optimizing $\boldsymbol{h}$ using (relaxed) PEP for first-order methods, and derives an analytical form of the optimized coefficients $\boldsymbol{h}$ and a corresponding new analytical bound. Section 7 investigates efficient formulations of the proposed first-order methods (OGM1 and OGM2). Section 8 shows that the corresponding analytical upper bound is tight and Section 9 concludes.

2 Problem and approach

2.1 Smooth convex minimization problem

We consider first-order algorithms for solving the following minimization problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} \ f(\boldsymbol{x}), \tag{M}$$

where the following two conditions are assumed:

- $f \ : \ \mathbb{R}^d \to \mathbb{R}$ is a convex function of the type $C_L^{1,1}(\mathbb{R}^d)$, *i.e.*, continuously differentiable with Lipschitz continuous gradient:

$$||f'(\boldsymbol{x}) - f'(\boldsymbol{y})|| \le L||\boldsymbol{x} - \boldsymbol{y}||, \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d,$$

  where $L > 0$ is the Lipschitz constant.
- The optimal set $X_*(f) = \arg\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x})$ is nonempty, *i.e.*, the problem (M) is solvable.

We focus on measuring the "inaccuracy" $f(\boldsymbol{x}_N) - f(\boldsymbol{x}_*)$ after N iterations to quantify the worst-case performance of any given first-order algorithm.

2.2 Optimizing the step coefficients $\boldsymbol{h}$ of first-order algorithms

In search of the best-performing first-order methods, DT [5] proposed to optimize $\boldsymbol{h} = \{h_{i,k}\}_{0\le k<i\le N}$ in Algorithm FO by minimizing a worst-case bound of $f(\boldsymbol{x}_N)-f(\boldsymbol{x}_*)$ for a given number of iterations $N \ge 1$ and initial distance $R > 0$, by adding $\arg\min_{\boldsymbol{h}}$ to problem (P) as follows:

$$\hat{\boldsymbol{h}}_{\mathrm{P}} \triangleq \mathop{\arg\min}_{\boldsymbol{h}\in\mathbb{R}^{N(N+1)/2}} \mathcal{B}_{\mathrm{P}}(\boldsymbol{h}, N, d, L, R). \tag{HP}$$

Note that $\hat{\boldsymbol{h}}_{\mathrm{P}}$ is independent[2] of L, R, and d. (See footnote 1.) Solving problem (HP) would give the step coefficients of the optimal first-order algorithm achieving the best worst-case convergence bound. DT [5] relaxed[3] problem (HP) to a tractable form, as reviewed in Sections 4 and 6.1. After these simplifications, the resulting solution was computed by a semidefinite program (SDP) that remains computationally expensive for large N [5]. In addition, the corresponding numerically optimized first-order algorithm was impractical for large-scale problems, requiring a linear combination of previous and current gradients $\{f'(\boldsymbol{x}_0), \cdots, f'(\boldsymbol{x}_i)\}$ at the $(i+1)$-th iteration.[4]

To make DT's work [5] practical, we directly derive the "analytical" solution for $\boldsymbol{h}$ in a relaxed version of the problem (HP), circumventing the numerical approach in [5]. Interestingly, the analytical solution of the relaxed version of (HP) satisfies a convenient recursion, so we provide practical optimized algorithms similar to Nesterov's efficient fast gradient methods.

3 Nesterov's fast gradient methods

This section reviews Nesterov's well-known fast gradient methods [10,12]. We further show the equivalence[5] of two of Nesterov's fast gradient methods in smooth unconstrained convex minimization. The analysis techniques used here will be important in Section 7.

3.1 Nesterov's fast gradient method 1 (FGM1)

Nesterov's first fast gradient method is called FGM1 [10]:

[2] Substituting $\boldsymbol{x}' = \frac{1}{R}\boldsymbol{x}$ and $\breve{f}(\boldsymbol{x}') = \frac{1}{LR^2} f(R\boldsymbol{x}') \in \mathcal{F}_1(\mathbb{R}^d)$ in problem (P), we get $\mathcal{B}_{\mathrm{P}}(\boldsymbol{h}, N, L, R) = LR^2 \mathcal{B}_{\mathrm{P}}(\boldsymbol{h}, N, 1, 1)$. This leads to $\hat{\boldsymbol{h}}_{\mathrm{P}} = \arg\min_{\boldsymbol{h}} \mathcal{B}_{\mathrm{P}}(\boldsymbol{h}, N, L, R) = \arg\min_{\boldsymbol{h}} \mathcal{B}_{\mathrm{P}}(\boldsymbol{h}, N, 1, 1)$.

[3] Using the term 'best' or 'optimal' here for [5] may be too strong, since [5] relaxed (HP) to a solvable form. We also use these relaxations, so we use the term "optimized" for our proposed algorithms.

[4] If coefficients $\boldsymbol{h}$ in Algorithm FO have a special recursive form, it is possible to find an equivalent efficient form, as discussed in Sections 3 and 7.

[5] The equivalence of two of Nesterov's fast gradient methods for smooth unconstrained convex minimization was previously mentioned without details in [18].

```
**Algorithm FGM1**
  Input: $f \in C_L^{1,1}(\mathbb{R}^d)$ convex, $\boldsymbol{x}_0 \in \mathbb{R}^d$, $\boldsymbol{y}_0 = \boldsymbol{x}_0$, $t_0 = 1$.
  For $i = 0, \cdots, N-1$
      $\boldsymbol{y}_{i+1} = \boldsymbol{x}_i - \frac{1}{L} f'(\boldsymbol{x}_i)$
      $t_{i+1} = \frac{1 + \sqrt{1 + 4t_i^2}}{2}$                                    (3.1)
      $\boldsymbol{x}_{i+1} = \boldsymbol{y}_{i+1} + \frac{t_i - 1}{t_{i+1}} (\boldsymbol{y}_{i+1} - \boldsymbol{y}_i)$.
```

Note that t_i in (3.1) satisfies the following relationships used frequently in later derivations:

$$t_{i+1}^2 - t_{i+1} - t_i^2 = 0, \quad t_i^2 = \sum_{k=0}^{i} t_k, \quad \text{and} \quad t_i \geq \frac{i+2}{2}, \quad i = 0, 1, \cdots . \tag{3.2}$$

Algorithm FGM1 is in Algorithm Class FO [5, Proposition 2] with:

$$\bar{h}_{i+1,k} = \begin{cases} \frac{t_i - 1}{t_{i+1}} \bar{h}_{i,k}, & k = 0, \cdots, i-2, \\ \frac{t_i - 1}{t_{i+1}} (\bar{h}_{i,i-1} - 1), & k = i-1, \\ 1 + \frac{t_i - 1}{t_{i+1}}, & k = i, \end{cases} \tag{3.3}$$

for $i = 0, \cdots, N-1$. Note that Algorithm FO with (3.3) is impractical as written for large-scale optimization problems, whereas the mathematically equivalent version FGM1 is far more useful practically due to its efficient form.

While the sequence $\{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_{N-1}, \boldsymbol{y}_N\}$ of FGM1 can be also written in class FO [5, Proposition 2], only the *primary* sequence $\{\boldsymbol{y}_0, \cdots, \boldsymbol{y}_N\}$ is known to achieve the rate $O\left(\frac{1}{N^2}\right)$ for decreasing f [2,10]. DT conjectured that the *secondary* sequence $\{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_N\}$ of FGM1 also achieves the same $O\left(\frac{1}{N^2}\right)$ rate based on the numerical results using the PEP approach [5, Conjecture 2]; our Section 5 verifies the conjecture by providing an analytical bound using the PEP approach.

3.2 Nesterov's fast gradient method 2 (FGM2)

In [12], Nesterov proposed another fast gradient method that has a different[6] form than FGM1 and that used a choice of t_i factors different from (3.1). Here, we use (3.1) because it leads to faster convergence than the factors used in [12]. The algorithm in [12] then becomes FGM2 shown below.

[6] The fast gradient method in [12] was originally developed to generalize FGM1 to the constrained case. Here, this second form is introduced for use in later proofs.

```
Algorithm FGM2
  Input: $f \in C_L^{1,1}(\mathbb{R}^d)$ convex, $\boldsymbol{x}_0 \in \mathbb{R}^d$, $t_0 = 1$.
  For $i = 0, \cdots, N-1$
      $\boldsymbol{y}_{i+1} = \boldsymbol{x}_i - \frac{1}{L} f'(\boldsymbol{x}_i)$
      $\boldsymbol{z}_{i+1} = \boldsymbol{x}_0 - \frac{1}{L} \sum_{k=0}^{i} t_k f'(\boldsymbol{x}_k)$
      $t_{i+1} = \frac{1 + \sqrt{1 + 4t_i^2}}{2}$
      $\boldsymbol{x}_{i+1} = \left(1 - \frac{1}{t_{i+1}}\right) \boldsymbol{y}_{i+1} + \frac{1}{t_{i+1}} \boldsymbol{z}_{i+1}$
```

Similar to FGM1, the following proposition shows that FGM2 is in class FO with

$$\bar{h}_{i+1,k} = \begin{cases} \frac{1}{t_{i+1}} \left(t_k - \sum_{j=k+1}^{i} \bar{h}_{j,k} \right), & k = 0, \cdots, i-1, \\ 1 + \frac{t_i - 1}{t_{i+1}}, & k = i, \end{cases} \tag{3.4}$$

for $i = 0, \cdots, N-1$ with t_i in (3.1).

Proposition 1 *The sequence $\{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_N\}$ generated by Algorithm FO with (3.4) is identical to the corresponding sequence generated by Algorithm FGM2.*

Proof We use induction, and for clarity, we use the notation $\boldsymbol{x}_0', \cdots, \boldsymbol{x}_N'$ for Algorithm FO. Clearly $\boldsymbol{x}_0' = \boldsymbol{x}_0$. To prove equivalence for $i = 1$:

$$\begin{aligned} \boldsymbol{x}_1' &= \boldsymbol{x}_0' - \frac{1}{L} \bar{h}_{1,0} f'(\boldsymbol{x}_0') = \boldsymbol{x}_0 - \frac{1}{L} \left(1 + \frac{t_0 - 1}{t_1}\right) f'(\boldsymbol{x}_0) \\ &= \left(1 - \frac{1}{t_1} + \frac{1}{t_1}\right) \left(\boldsymbol{x}_0 - \frac{1}{L} f'(\boldsymbol{x}_0)\right) = \left(1 - \frac{1}{t_1}\right) \boldsymbol{y}_1 + \frac{1}{t_1} \boldsymbol{z}_1 = \boldsymbol{x}_1. \end{aligned}$$

Assuming $\boldsymbol{x}_i' = \boldsymbol{x}_i$ for $i = 0, \cdots, n$, we then have

$$\begin{aligned} \boldsymbol{x}_{n+1}' =& \boldsymbol{x}_n' - \frac{1}{L} \bar{h}_{n+1,n} f'(\boldsymbol{x}_n') - \frac{1}{L} \sum_{k=0}^{n-1} \bar{h}_{n+1,k} f'(\boldsymbol{x}_k') \\ =& \boldsymbol{x}_n - \frac{1}{L} \left(1 + \frac{t_n - 1}{t_{n+1}}\right) f'(\boldsymbol{x}_n) - \frac{1}{L} \sum_{k=0}^{n-1} \frac{1}{t_{n+1}} \left(t_k - \sum_{j=k+1}^{n} \bar{h}_{j,k} \right) f'(\boldsymbol{x}_k) \\ =& \left(1 - \frac{1}{t_{n+1}}\right) \left(\boldsymbol{x}_n - \frac{1}{L} f'(\boldsymbol{x}_n)\right) \\ &+ \frac{1}{t_{n+1}} \left(\boldsymbol{x}_n + \frac{1}{L} \sum_{k=0}^{n-1} \sum_{j=k+1}^{n} \bar{h}_{j,k} f'(\boldsymbol{x}_k) - \frac{1}{L} \sum_{k=0}^{n} t_k f'(\boldsymbol{x}_k) \right) \end{aligned}$$

$$= \left(1 - \frac{1}{t_{n+1}}\right) \boldsymbol{y}_{n+1} + \frac{1}{t_{n+1}} \left(\boldsymbol{x}_n + \frac{1}{L} \sum_{j=1}^{n} \sum_{k=0}^{j-1} \bar{h}_{j,k} f'(\boldsymbol{x}_k) - \frac{1}{L} \sum_{k=0}^{n} t_k f'(\boldsymbol{x}_k) \right)$$
$$= \left(1 - \frac{1}{t_{n+1}}\right) \boldsymbol{y}_{n+1} + \frac{1}{t_{n+1}} \left(\boldsymbol{x}_0 - \frac{1}{L} \sum_{k=0}^{n} t_k f'(\boldsymbol{x}_k) \right) = \boldsymbol{x}_{n+1}.$$

The fifth equality uses the telescoping sum $\boldsymbol{x}_n = \boldsymbol{x}_0 + \sum_{j=1}^{n}(\boldsymbol{x}_j - \boldsymbol{x}_{j-1})$ and (1.1) in Algorithm FO. □

We show next the equivalence of Nesterov's two algorithms FGM1 and FGM2 for smooth unconstrained convex minimization using (3.3) and (3.4).

Proposition 2 *The sequence $\{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_N\}$ generated by Algorithm FGM2 is identical to the corresponding sequence generated by Algorithm FGM1.*

Proof *We prove the statement by showing the equivalence of (3.3) and (3.4). We use the notation $\bar{h}'_{i,k}$ for the coefficients (3.4) of Algorithm FGM2 to distinguish from those of Algorithm FGM1.*

It is obvious that $\bar{h}'_{i+1,i} = \bar{h}_{i+1,i},\ i = 0, \cdots, N-1$, and we can easily prove for $i = 0, \cdots, N-1$ that

$$\bar{h}'_{i+1,i-1} = \frac{1}{t_{i+1}}\left(t_{i-1} - \bar{h}'_{i,i-1}\right) = \frac{1}{t_{i+1}}\left(t_{i-1} - \left(1 + \frac{t_{i-1} - 1}{t_i}\right)\right)$$
$$= \frac{(t_i - 1)(t_{i-1} - 1)}{t_i t_{i+1}} = \frac{t_i - 1}{t_{i+1}}\left(\bar{h}_{i,i-1} - 1\right) = \bar{h}_{i+1,i-1}.$$

We next use induction by assuming $\bar{h}'_{i+1,k} = \bar{h}_{i+1,k}$ for $i = 0, \cdots, n-1,\ k = 0, \cdots, i$. We then have

$$\bar{h}'_{n+1,k} = \frac{1}{t_{n+1}}\left(t_k - \sum_{j=k+1}^{n} \bar{h}'_{j,k}\right) = \frac{1}{t_{n+1}}\left(t_k - \sum_{j=k+1}^{n-1} \bar{h}'_{j,k} - \bar{h}'_{n,k}\right)$$
$$= \frac{t_n - 1}{t_{n+1}} \bar{h}'_{n,k} = \frac{t_n - 1}{t_{n+1}} \bar{h}_{n,k} = \bar{h}_{n+1,k}$$

for $k = 0, \cdots, n-2$. Note that this proof is independent of the choice of t_i. □

3.3 A convergence bound for Nesterov's fast gradient methods

Algorithms FGM1 and FGM2 generate the same sequences $\{\boldsymbol{x}_i\}$ and $\{\boldsymbol{y}_i\}$, and the primary sequence $\{\boldsymbol{y}_i\}$ is known to satisfy the bound[7] [2,10,12]:

$$f(\boldsymbol{y}_n) - f(\boldsymbol{x}_*) \leq \frac{L||\boldsymbol{x}_0 - \boldsymbol{x}_*||^2}{2t_{n-1}^2} \leq \frac{2L||\boldsymbol{x}_0 - \boldsymbol{x}_*||^2}{(n+1)^2}, \quad \forall \boldsymbol{x}_* \in X_*(f) \tag{3.5}$$

[7] The second inequality of (3.5) is widely known since it provides simpler interpretation of a convergence bound, compared to the first inequality of (3.5).

for $n \geq 1$, which was the previously best known analytical bound of first-order methods for smooth unconstrained convex minimization; DT's PEP approach provides a tighter *numerical* bound for the sequences $\{\boldsymbol{x}_i\}$ and $\{\boldsymbol{y}_i\}$ compared to the analytical bound (3.5) [5, Table 1]. Using the PEP approach, Section 5 provides a new *analytical* bound for the secondary sequence $\{\boldsymbol{x}_i\}$ of FGM1 and FGM2.

Nesterov described a convex function $f \in C_L^{1,1}(\mathbb{R}^d)$ for which any first-order algorithm generating the sequence $\{\boldsymbol{x}_i\}$ in the class of Algorithm FO satisfies [11, Theorem 2.1.7]:

$$\frac{3L||\boldsymbol{x}_0 - \boldsymbol{x}_*||^2}{32(n+1)^2} \leq f(\boldsymbol{x}_n) - f(\boldsymbol{x}_*), \quad \forall \boldsymbol{x}_* \in X_*(f) \tag{3.6}$$

for $n = 1, \cdots, \left\lfloor \frac{d-1}{2} \right\rfloor$, indicating that Nesterov's two FGM1 and FMG2 achieve the optimal rate $O\left(\frac{1}{N^2}\right)$. (Note that the bound (3.6) is valid if the large-scale condition "$d \geq 2N+1$" is satisfied.) However, (3.6) also illustrates the potential room for improving first-order algorithms by a constant factor.

To narrow this gap, DT [5] used a relaxation of problem (HP) to find the "optimal" choice of $\{h_{i,k}\}$ for Algorithm FO that minimizes a relaxed bound on $f(\boldsymbol{x}_N) - f(\boldsymbol{x}_*)$ at the Nth iteration, which was found numerically to provide a twice smaller bound than (3.5), yet remained computationally impractical.

We next review the PEP approach for solving a relaxed version of (P).

4 DT's convergence bound for first-order algorithms using PEP

This section summarizes the relaxation scheme for the PEP approach that transforms problem (P) into a tractable form [5]. The relaxed PEP bounds are used in later sections.

Problem (P) is challenging to solve due to the (infinite-dimensional) functional constraint on f, so DT [5] cleverly relax the constraint by using a well-known property for the class of convex $C_L^{1,1}$ functions in [11, Theorem 2.1.5] and further relax as follows:

$$\begin{aligned}\mathcal{B}_{\text{P1}}(\boldsymbol{h}, N, d, L, R) \triangleq \; & \max_{\substack{\boldsymbol{G} \in \mathbb{R}^{(N+1)d}, \\ \boldsymbol{\delta} \in \mathbb{R}^{N+1}}} LR^2 \delta_N \\ & \text{s.t. } \operatorname{Tr}\left\{\boldsymbol{G}^\top \boldsymbol{A}_{i-1,i}(\boldsymbol{h}) \boldsymbol{G}\right\} \leq \delta_{i-1} - \delta_i, \quad i = 1, \cdots, N, \\ & \phantom{\text{s.t. }} \operatorname{Tr}\left\{\boldsymbol{G}^\top \boldsymbol{D}_i(\boldsymbol{h}) \boldsymbol{G} + \boldsymbol{\nu} \boldsymbol{u}_i^\top \boldsymbol{G}\right\} \leq -\delta_i, \quad i = 0, \cdots, N,\end{aligned} \tag{P1}$$

for any given unit vector $\boldsymbol{\nu} \in \mathbb{R}^d$, by defining $\delta_i \triangleq \frac{1}{L||\boldsymbol{x}_0 - \boldsymbol{x}_*||^2}(f(\boldsymbol{x}_i) - f(\boldsymbol{x}_*))$ and $\boldsymbol{g}_i \triangleq \frac{1}{L||\boldsymbol{x}_0 - \boldsymbol{x}_*||} f'(\boldsymbol{x}_i)$ for $i = 0, \cdots, N, *$, and denoting the unit vectors[8]

[8] The vector $\boldsymbol{e}_{N,i}$ is the ith standard basis vector in $\mathbb{R}^N$, having 1 for the ith entry and zero for all other elements.

$\boldsymbol{u}_i = \boldsymbol{e}_{N+1,i+1} \in \mathbb{R}^{N+1}$, the $(N+1)\times 1$ vector $\boldsymbol{\delta} = [\delta_0, \ \cdots, \ \delta_N]^\top$, the $(N+1)\times d$ matrix $\boldsymbol{G} = [\boldsymbol{g}_0, \ \cdots, \ \boldsymbol{g}_N]^\top$, and the $(N+1)\times(N+1)$ symmetric matrices:

$$\begin{cases} \boldsymbol{A}_{i-1,i}(\boldsymbol{h}) \triangleq \frac{1}{2}(\boldsymbol{u}_{i-1}-\boldsymbol{u}_i)(\boldsymbol{u}_{i-1}-\boldsymbol{u}_i)^\top + \frac{1}{2}\sum_{k=0}^{i-1} h_{i,k}(\boldsymbol{u}_i\boldsymbol{u}_k^\top + \boldsymbol{u}_i\boldsymbol{u}_j^\top), \\ \boldsymbol{D}_i(\boldsymbol{h}) \triangleq \frac{1}{2}\boldsymbol{u}_i\boldsymbol{u}_i^\top + \frac{1}{2}\sum_{j=1}^{i}\sum_{k=0}^{j-1} h_{j,k}(\boldsymbol{u}_i\boldsymbol{u}_k^\top + \boldsymbol{u}_k\boldsymbol{u}_i^\top). \end{cases} \tag{4.1}$$

DT [5] finally use a duality approach on (P1). Replacing $\max_{\boldsymbol{G},\boldsymbol{\delta}} LR^2\delta_N$ by $\min_{\boldsymbol{G},\boldsymbol{\delta}} -\delta_N$ for convenience, the Lagrangian of the corresponding constrained minimization problem (P1) becomes the following separable function in $(\boldsymbol{\delta}, \boldsymbol{G})$:

$$\mathcal{L}(\boldsymbol{G},\boldsymbol{\delta},\boldsymbol{\lambda},\boldsymbol{\tau};\boldsymbol{h}) = \mathcal{L}_1(\boldsymbol{\delta},\boldsymbol{\lambda},\boldsymbol{\tau}) + \mathcal{L}_2(\boldsymbol{G},\boldsymbol{\lambda},\boldsymbol{\tau};\boldsymbol{h}),$$

where

$$\begin{cases} \mathcal{L}_1(\boldsymbol{\delta},\boldsymbol{\lambda},\boldsymbol{\tau}) \triangleq -\delta_N + \sum_{i=1}^N \lambda_i(\delta_i - \delta_{i-1}) + \sum_{i=0}^N \tau_i\delta_i, \\ \mathcal{L}_2(\boldsymbol{G},\boldsymbol{\lambda},\boldsymbol{\tau};\boldsymbol{h}) \triangleq \sum_{i=1}^N \lambda_i \operatorname{Tr}\{\boldsymbol{G}^\top\boldsymbol{A}_{i-1,i}(\boldsymbol{h})\boldsymbol{G}\} + \sum_{i=0}^N \tau_i \operatorname{Tr}\{\boldsymbol{G}^\top\boldsymbol{D}_i(\boldsymbol{h})\boldsymbol{G} + \boldsymbol{\nu}\boldsymbol{u}_i^\top\boldsymbol{G}\}, \end{cases}$$

with dual variables $\boldsymbol{\lambda} = (\lambda_1,\cdots,\lambda_N)^\top \in \mathbb{R}_+^N$ and $\boldsymbol{\tau} = (\tau_0,\cdots,\tau_N)^\top \in \mathbb{R}_+^{N+1}$. The corresponding dual function is defined as

$$H(\boldsymbol{\lambda},\boldsymbol{\tau};\boldsymbol{h}) = \min_{\boldsymbol{\delta}\in\mathbb{R}^{N+1}} \mathcal{L}_1(\boldsymbol{\delta},\boldsymbol{\lambda},\boldsymbol{\tau}) + \min_{\boldsymbol{G}\in\mathbb{R}^{(N+1)d}} \mathcal{L}_2(\boldsymbol{G},\boldsymbol{\lambda},\boldsymbol{\tau};\boldsymbol{h}). \tag{4.2}$$

Here $\min_{\boldsymbol{\delta}} \mathcal{L}_1(\boldsymbol{\delta},\boldsymbol{\lambda},\boldsymbol{\tau}) = 0$ for any $(\boldsymbol{\lambda},\boldsymbol{\tau}) \in \Lambda$, where

$$\Lambda = \left\{ (\boldsymbol{\lambda},\boldsymbol{\tau}) \in \mathbb{R}_+^N \times \mathbb{R}_+^{N+1} \ : \ \begin{array}{l} \tau_0 = \lambda_1, \ \lambda_N + \tau_N = 1 \\ \lambda_i - \lambda_{i+1} + \tau_i = 0, \ i = 1,\cdots,N-1 \end{array} \right\}, \tag{4.3}$$

and $-\infty$ otherwise. In [5], the dual function (4.2) for any given unit vector $\boldsymbol{\nu} \in \mathbb{R}^d$ was found to be

$$\begin{aligned} H(\boldsymbol{\lambda},\boldsymbol{\tau};\boldsymbol{h}) &= \min_{\boldsymbol{w}\in\mathbb{R}^{N+1}} \{\boldsymbol{w}^\top\boldsymbol{S}(\boldsymbol{h},\boldsymbol{\lambda},\boldsymbol{\tau})\boldsymbol{w} + \boldsymbol{\tau}^\top\boldsymbol{w}\} \\ &= \max_{\gamma\in\mathbb{R}} \left\{ -\frac{1}{2}\gamma \ : \ \boldsymbol{w}^\top\boldsymbol{S}(\boldsymbol{h},\boldsymbol{\lambda},\boldsymbol{\tau})\boldsymbol{w} + \boldsymbol{\tau}^\top\boldsymbol{w} \ge -\frac{1}{2}\gamma, \ \forall \boldsymbol{w} \in \mathbb{R}^{N+1} \right\} \\ &= \max_{\gamma\in\mathbb{R}} \left\{ -\frac{1}{2}\gamma \ : \ \begin{pmatrix} \boldsymbol{S}(\boldsymbol{h},\boldsymbol{\lambda},\boldsymbol{\tau}) & \frac{1}{2}\boldsymbol{\tau} \\ \frac{1}{2}\boldsymbol{\tau}^\top & \frac{1}{2}\gamma \end{pmatrix} \succeq 0 \right\} \end{aligned} \tag{4.4}$$

for any given $(\boldsymbol{\lambda},\boldsymbol{\tau}) \in \Lambda$, where DT [5] define the following $(N+1)\times(N+1)$ matrix using the definition of $\boldsymbol{A}_{i-1,i}(\boldsymbol{h})$ and $\boldsymbol{D}_i(\boldsymbol{h})$ in (4.1):

$$\begin{aligned} \boldsymbol{S}(\boldsymbol{h},\boldsymbol{\lambda},\boldsymbol{\tau}) =& \sum_{i=1}^N \lambda_i\boldsymbol{A}_{i-1,i}(\boldsymbol{h}) + \sum_{i=0}^N \tau_i\boldsymbol{D}_i(\boldsymbol{h}) \\ =& \frac{1}{2}\sum_{i=1}^N \lambda_i(\boldsymbol{u}_{i-1}-\boldsymbol{u}_i)(\boldsymbol{u}_{i-1}-\boldsymbol{u}_i)^\top + \frac{1}{2}\sum_{i=0}^N \tau_i\boldsymbol{u}_i\boldsymbol{u}_i^\top \\ &+ \frac{1}{2}\sum_{i=1}^N\sum_{k=0}^{i-1}\left(\lambda_i h_{i,k} + \tau_i \sum_{j=k+1}^{i} h_{j,k}\right)(\boldsymbol{u}_i\boldsymbol{u}_k^\top + \boldsymbol{u}_k\boldsymbol{u}_i^\top). \end{aligned} \tag{4.5}$$

In short, using the dual approach on the problem (P1) yields the following bound:

$$\mathcal{B}_{\mathrm{D}}(\boldsymbol{h}, N, L, R) \triangleq \min_{\substack{\boldsymbol{\lambda}\in\mathbb{R}^N,\\ \boldsymbol{\tau}\in\mathbb{R}^{N+1},\\ \gamma\in\mathbb{R}}} \left\{ \frac{1}{2}LR^2\gamma \; : \; \begin{pmatrix} \boldsymbol{S}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau}) & \frac{1}{2}\boldsymbol{\tau} \\ \frac{1}{2}\boldsymbol{\tau}^\top & \frac{1}{2}\gamma \end{pmatrix} \succeq 0, \quad (\boldsymbol{\lambda}, \boldsymbol{\tau}) \in \Lambda \right\}, \tag{D}$$

recalling that we previously replaced $\max_{\boldsymbol{G},\boldsymbol{\lambda}} LR^2\delta_N$ by $\min_{\boldsymbol{G},\boldsymbol{\lambda}} -\delta_N$ for convenience. Problem (D) can be solved using any numerical SDP method [3,6] for given $\boldsymbol{h}$ and N, noting that R is just a multiplicative scalar in (D). Interestingly, this bound $\mathcal{B}_{\mathrm{D}}(\boldsymbol{h}, N, L, R)$ is independent of dimension d.

Overall, DT [5] introduced a series of relaxations to the problem (P), eventually reaching the solvable problem (D) that provides a valid upper bound as

$$f(\boldsymbol{x}_N) - f(\boldsymbol{x}_*) \leq \mathcal{B}_{\mathrm{P}}(\boldsymbol{h}, N, d, L, R) \leq \mathcal{B}_{\mathrm{D}}(\boldsymbol{h}, N, L, R)$$

where $\boldsymbol{x}_N$ is generated by Algorithm FO with given $\boldsymbol{h}$ and N, and $||\boldsymbol{x}_0 - \boldsymbol{x}_*|| \leq R$. This bound is for a given $\boldsymbol{h}$ and later we optimize the bound over $\boldsymbol{h}$.

Solving problem (D) with a SDP method for any given coefficients $\boldsymbol{h}$ and N provides a numerical convergence bound for $f(\boldsymbol{x}_N) - f(\boldsymbol{x}_*)$ [5]. However, numerical bounds only partially explain the behavior of algorithms in class FO. An analytical bound of gradient methods with a constant step $0 < h \leq 1$, for example, was found using a specific PEP approach [5], but no other analytical bound was discussed in [5]. The next section exploits the PEP approach to reveal a new analytical bound for the secondary sequence $\{f(\boldsymbol{x}_i)\}$ generated by FGM1 or FGM2 as an example, confirming the conjecture by DT that the *secondary* sequence $\{\boldsymbol{x}_i\}$ achieves the same rate $O\left(\frac{1}{N^2}\right)$ as the *primary* sequence $\{\boldsymbol{y}_i\}$ [5, Conjecture 2].

5 A new analytical bound for Nesterov's fast gradient methods

This section provides an analytical bound for the secondary sequence $\{\boldsymbol{x}_i\}$ in FGM1 and FGM2.

For the $\bar{\boldsymbol{h}}$ factors in (3.3) or (3.4) of Nesterov's fast gradient methods, the following choice of dual variables (inspired by Section 6.2) is a feasible point of problem (D):

$$\bar{\lambda}_i = \frac{t_{i-1}^2}{t_N^2}, \quad i = 1, \cdots, N, \tag{5.1}$$

$$\bar{\tau}_i = \frac{t_i}{t_N^2}, \quad i = 0, \cdots, N, \tag{5.2}$$

$$\bar{\gamma} = \frac{1}{t_N^2}, \tag{5.3}$$

with t_i in (3.1), as shown in the following lemma.

Lemma 1 *The choice* $(\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\tau}}, \bar{\gamma})$ *in* (5.1), (5.2) *and* (5.3) *is a feasible point of the problem* (D) *for the* $\bar{\boldsymbol{h}}$ *designs given in* (3.3) *or* (3.4) *that are used in Nesterov's FGM1 and FGM2.*

Proof *It is obvious that* $(\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\tau}}) \in \Lambda$ *using* $t_i^2 = \sum_{k=0}^{i} t_k$ *in* (3.2). *We next rewrite* $\boldsymbol{S}(\bar{\boldsymbol{h}}, \bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\tau}})$ *using* (3.4), (5.1) *and* (5.2) *to show that the choice* $(\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\tau}}, \bar{\gamma})$ *satisfies the positive semidefinite condition in* (D) *for given* $\bar{\boldsymbol{h}}$.

For any $\boldsymbol{h}$ *and* $(\boldsymbol{\lambda}, \boldsymbol{\tau}) \in \Lambda$, *the* (i,k)*-th entry of the symmetric matrix* $\boldsymbol{S}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau})$ *in* (4.5) *can be written as*

$$S_{i,k}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau}) = \begin{cases} \frac{1}{2}\Big((\lambda_i + \tau_i)h_{i,k} + \tau_i \sum_{j=k+1}^{i-1} h_{j,k}\Big), & \\ & i = 2, \cdots, N,\ k = 0, \cdots, i-2, \\ \frac{1}{2}\left((\lambda_i + \tau_i)h_{i,k} - \lambda_i\right), & i = 1, \cdots, N,\ k = i-1, \\ \lambda_{i+1}, & i = 0, \cdots, N-1,\ k = i, \\ \frac{1}{2}, & i = N,\ k = i. \end{cases} \tag{5.4}$$

Inserting $\bar{\boldsymbol{h}}$ (3.4), $\bar{\boldsymbol{\lambda}}$ (5.1) *and* $\bar{\boldsymbol{\tau}}$ (5.2) *into* (5.4) *and using* $\bar{\lambda}_i + \bar{\tau}_i = \frac{t_i^2}{t_N^2}$ *for* $i = 1, \cdots, N$, *we get*

$$\begin{aligned} S_{i,k}(\bar{\boldsymbol{h}}, \bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\tau}}) &= \begin{cases} \frac{1}{2}\Big(\frac{t_i^2}{t_N^2}\frac{1}{t_i}\Big(t_k - \sum_{j=k+1}^{i-1} \bar{h}_{j,k}\Big) + \frac{t_i}{t_N^2}\sum_{j=k+1}^{i-1} \bar{h}_{j,k}\Big), & \\ & i = 2, \cdots, N,\ k = 0, \cdots, i-2, \\ \frac{1}{2}\left(\frac{t_i^2}{t_N^2}\left(1 + \frac{t_{i-1}-1}{t_i}\right) - \frac{t_{i-1}^2}{t_N^2}\right), & i = 1, \cdots, N,\ k = i-1, \\ \frac{t_i^2}{t_N^2}, & i = 0, \cdots, N-1,\ k = i, \\ \frac{1}{2}, & i = N,\ k = i, \end{cases} \\ &= \begin{cases} \frac{t_i t_k}{2t_N^2} & i = 1, \cdots, N,\ k = 0, \cdots, i-1, \\ \frac{t_i^2}{t_N^2}, & i = 0, \cdots, N-1,\ k = i, \\ \frac{t_N^2}{2t_N^2}, & i = N,\ k = i, \end{cases} \\ &= \frac{1}{2t_N^2}\left(\boldsymbol{t}\,\boldsymbol{t}^\top + \mathsf{diag}\{(\check{\boldsymbol{t}}^\top, 0)\}\right), \end{aligned}$$

where $\boldsymbol{t} = (t_0, \cdots, t_N)^\top$ *and* $\check{\boldsymbol{t}} = (t_0^2, \cdots, t_{N-1}^2)^\top$. *The second equality uses* $t_i^2 - t_i - t_{i-1}^2 = 0$ *in* (3.2), *and* $\mathsf{diag}\{\boldsymbol{t}\}$ *denotes a matrix where diagonal elements are filled with elements of a vector* $\boldsymbol{t}$ *and zero for other elements.*

Finally, using $\bar{\gamma}$ *in* (5.3), *we have*

$$\begin{aligned} \begin{pmatrix} \boldsymbol{S}(\bar{\boldsymbol{h}}, \bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\tau}}) & \frac{1}{2}\bar{\boldsymbol{\tau}} \\ \frac{1}{2}\bar{\boldsymbol{\tau}}^\top & \frac{1}{2}\bar{\gamma} \end{pmatrix} &= \begin{pmatrix} \frac{1}{2t_N^2}\left(\boldsymbol{t}\,\boldsymbol{t}^\top + \mathsf{diag}\{(\check{\boldsymbol{t}}^\top, 0)\}\right) & \frac{1}{2t_N^2}\boldsymbol{t} \\ \frac{1}{2t_N^2}\boldsymbol{t}^\top & \frac{1}{2t_N^2} \end{pmatrix} \\ &= \frac{1}{2t_N^2}\left\{\begin{pmatrix} \boldsymbol{t} \\ 1 \end{pmatrix}\begin{pmatrix} \boldsymbol{t} \\ 1 \end{pmatrix}^\top + \mathsf{diag}\{(\check{\boldsymbol{t}}^\top, 0, 0)\}\right\} \succeq 0. \end{aligned}$$

□

Using Lemma 1, we provide an analytical convergence bound for the secondary sequence $\{\boldsymbol{x}_i\}$ of FGM1 and FMG2.

Theorem 1 *Let $f : \mathbb{R}^d \to \mathbb{R}$ be convex and $C_L^{1,1}$ and let $\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots \in \mathbb{R}^d$ be generated by FGM1 or FGM2. Then for $n \geq 1$,*

$$f(\boldsymbol{x}_n) - f(\boldsymbol{x}_*) \leq \frac{L||\boldsymbol{x}_0 - \boldsymbol{x}_*||^2}{2t_n^2} \leq \frac{2L||\boldsymbol{x}_0 - \boldsymbol{x}_*||^2}{(n+2)^2}, \quad \forall \boldsymbol{x}_* \in X_*(f). \tag{5.5}$$

Proof *Using $\bar{\gamma}$ (5.3) and $t_N^2 \geq \frac{(N+2)^2}{4}$ from (3.2), we have*

$$f(\boldsymbol{x}_N) - f(\boldsymbol{x}_*) \leq \mathcal{B}_{\mathrm{D}}(\bar{\boldsymbol{h}}, N, L, R) \leq \frac{1}{2}LR^2\bar{\gamma} \leq \frac{2LR^2}{(N+2)^2}, \quad \forall \boldsymbol{x}_* \in X_*(f) \tag{5.6}$$

for given $\bar{\boldsymbol{h}}$ in (3.3) or (3.4), based on Lemma 1. Since the coefficients $\bar{\boldsymbol{h}}$ in (3.3) or (3.4) are recursive and do not depend on a given N, we can extend (5.6) for all iterations ($n \geq 1$). Finally, we let $R = ||\boldsymbol{x}_0 - \boldsymbol{x}_*||$. $\square$

Theorem 1 illustrates using the PEP approach to find an analytical bound for an algorithm in class FO. We used a SDP solver [3,6] to verify numerically that the choice $(\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\tau}}, \bar{\gamma})$ in (5.1), (5.2) and (5.3) is not an optimal solution of (D) for given $\bar{\boldsymbol{h}}$ in (3.3) or (3.4). Nevertheless, this feasible point $(\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\tau}}, \bar{\gamma})$ provides a valid upper bound for the sequence $\{\boldsymbol{x}_i\}$ of FGM1 and FGM2 as shown in Theorem 1 that is similar to (3.5) and verifies DT's conjecture [5, Conjecture 2].

The next section reviews DT's work [5] on numerically optimizing step coefficients $\boldsymbol{h}$ in the class of first-order methods over the relaxed convergence bound $\mathcal{B}_{\mathrm{D}}(\boldsymbol{h}, N, L, R)$. Then, we find an analytical form of the optimized step coefficients and explicitly show that Algorithm FO with such coefficients achieves a convergence bound that is twice as small as (3.5) and (5.5).

6 Towards optimized first-order algorithms

6.1 DT's numerically optimized first-order algorithms

This section summarizes the numerically optimized first-order algorithms described in [5].

Having relaxed (P) in Section 4 to (D), DT proposed to optimize $\boldsymbol{h}$ by relaxing (HP) as follows:

$$\hat{\boldsymbol{h}} \triangleq \operatorname*{arg\,min}_{\boldsymbol{h} \in \mathbb{R}^{N(N+1)/2}} \mathcal{B}_{\mathrm{D}}(\boldsymbol{h}, N, L, R), \tag{HD}$$

where $\hat{\boldsymbol{h}}$ is independent of both L and R, since $\mathcal{B}_{\mathrm{D}}(\boldsymbol{h}, N, L, R) = LR^2\mathcal{B}_{\mathrm{D}}(\boldsymbol{h}, N, 1, 1)$. Problem (HD) is a bilinear optimization problem in terms of $\boldsymbol{h}$ and the dual variables in (D), unlike the linear SDP problem (D). To simplify, DT [5] introduced a variable $\boldsymbol{r} = \{r_{i,k}\}_{0\le k<i\le N}$:

$$r_{i,k} = \lambda_i h_{i,k} + \tau_i \sum_{j=k+1}^{i} h_{j,k} \tag{6.1}$$

to convert (HD) into the following linear SDP problem:

$$\hat{\boldsymbol{r}} \triangleq \operatorname*{arg\,min}_{\boldsymbol{r}\in\mathbb{R}^{N(N+1)/2}} \breve{\mathcal{B}}_{\mathrm{D}}(\boldsymbol{r}, N, L, R), \tag{RD}$$

where

$$\breve{\mathcal{B}}_{\mathrm{D}}(\boldsymbol{r}, N, L, R) \triangleq \min_{\substack{\boldsymbol{\lambda}\in\mathbb{R}^N,\\ \boldsymbol{\tau}\in\mathbb{R}^{N+1},\\ \gamma\in\mathbb{R}}} \left\{ \frac{1}{2}LR^2\gamma \;:\; \begin{pmatrix} \breve{\boldsymbol{S}}(\boldsymbol{r},\boldsymbol{\lambda},\boldsymbol{\tau}) & \frac{1}{2}\boldsymbol{\tau} \\ \frac{1}{2}\boldsymbol{\tau}^\top & \frac{1}{2}\gamma \end{pmatrix} \succeq 0, (\boldsymbol{\lambda},\boldsymbol{\tau}) \in \Lambda \right\},$$

$$\begin{aligned} \breve{\boldsymbol{S}}(\boldsymbol{r},\boldsymbol{\lambda},\boldsymbol{\tau}) \triangleq & \frac{1}{2}\sum_{i=1}^{N} \lambda_i(\boldsymbol{u}_{i-1}-\boldsymbol{u}_i)(\boldsymbol{u}_{i-1}-\boldsymbol{u}_i)^\top + \frac{1}{2}\sum_{i=0}^{N}\tau_i\boldsymbol{u}_i\boldsymbol{u}_i^\top \\ & + \frac{1}{2}\sum_{i=1}^{N}\sum_{k=0}^{i-1} r_{i,k}(\boldsymbol{u}_i\boldsymbol{u}_k^\top + \boldsymbol{u}_k\boldsymbol{u}_i^\top). \end{aligned} \tag{6.2}$$

An optimal solution $(\hat{\boldsymbol{r}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}}, \hat{\gamma})$ of (RD) for a given N can be computed by any numerical SDP method [3,6]. DT showed that the resulting values $(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}}, \hat{\gamma})$ with the following $\hat{\boldsymbol{h}}$:

$$\hat{h}_{i,k} = \begin{cases} \frac{\hat{r}_{i,k} - \hat{\tau}_i \sum_{j=k+1}^{i-1} \hat{h}_{j,k}}{\hat{\lambda}_i + \hat{\tau}_i}, & \hat{\lambda}_i + \hat{\tau}_i \neq 0 \\ 0, & \text{otherwise}, \end{cases} \tag{6.3}$$

for $i = 1, \cdots, N,\; k = 0, \cdots, i-1$ become an optimal solution of (HD) [5, Theorem 3],[9] where both (HD) and (RD) achieve the same optimal value, *i.e.*, $\mathcal{B}_{\mathrm{D}}(\hat{\boldsymbol{h}}, N, L, R) = \breve{\mathcal{B}}_{\mathrm{D}}(\hat{\boldsymbol{r}}, N, L, R)$.

The numerical results for problem (HD) in [5] provided a convergence bound that is about two-times smaller than that of Nesterov's fast gradient methods for a couple of choices of N in [5, Tables 1 and 2]. However, numerical calculations cannot verify the acceleration for all N, and SDP computation for solving (RD) becomes expensive for large N. In the next section, we analytically solve problem (HD), which is our first main contribution.

[9] Equation (5.2) in [5, Theorem 3] that is derived from (6.1) has typos that we fixed in (6.3).

6.2 Proposed analytically optimized first-order algorithms

This section provides an analytical optimal solution of (HD) by reformulating (RD).

We first find an equivalent form of the dual function $H(\boldsymbol{\lambda}, \boldsymbol{\tau}; \boldsymbol{h})$ in (4.2) that differs from (4.4) by using the following equality:

$$S_{N,N}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau}) = \frac{1}{2} \text{ for any } (\boldsymbol{\lambda}, \boldsymbol{\tau}) \in \Lambda, \tag{6.4}$$

i.e., the (N, N)-th entry of $\boldsymbol{S}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau})$ in (4.5) and (5.4) is $\frac{1}{2}$ for any $(\boldsymbol{\lambda}, \boldsymbol{\tau}) \in \Lambda$. Hereafter we use the notation

$$\boldsymbol{S}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau}) \triangleq \begin{pmatrix} \boldsymbol{Q}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau}) & \boldsymbol{q}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau}) \\ \boldsymbol{q}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau})^\top & \frac{1}{2} \end{pmatrix}, \quad \boldsymbol{w} \triangleq \begin{pmatrix} \breve{\boldsymbol{w}} \\ w_N \end{pmatrix}, \quad \text{and} \quad \boldsymbol{\tau} \triangleq \begin{pmatrix} \breve{\boldsymbol{\tau}} \\ \tau_N \end{pmatrix}, \tag{6.5}$$

where $\boldsymbol{Q}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau})$ is a $N \times N$ symmetric matrix, $\boldsymbol{q}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau})$, $\breve{\boldsymbol{w}}$ and $\breve{\boldsymbol{\tau}}$ are $N \times 1$ vectors, and w_N and τ_N are scalars. We omit the arguments $(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau})$ in $\boldsymbol{Q}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau})$ and $\boldsymbol{q}(\boldsymbol{h}, \boldsymbol{\lambda}, \boldsymbol{\tau})$ for notational simplicity in the next derivation. For any given $(\boldsymbol{\lambda}, \boldsymbol{\tau}) \in \Lambda$, we rewrite $H(\boldsymbol{\lambda}, \boldsymbol{\tau}; \boldsymbol{h})$ in (4.2) and (4.4) as follows:

$$\begin{aligned} H(\boldsymbol{\lambda}, \boldsymbol{\tau}; \boldsymbol{h}) &= \min_{\boldsymbol{w} \in \mathbb{R}^{N+1}} \left\{ \breve{\boldsymbol{w}}^\top \boldsymbol{Q} \breve{\boldsymbol{w}} + \breve{\boldsymbol{\tau}}^\top \breve{\boldsymbol{w}} + 2\breve{\boldsymbol{w}}^\top \boldsymbol{q} w_N + \frac{1}{2} w_N^2 + \tau_N w_N \right\} \\ &= \min_{\breve{\boldsymbol{w}} \in \mathbb{R}^N} \left\{ \breve{\boldsymbol{w}}^\top (\boldsymbol{Q} - 2\boldsymbol{q}\boldsymbol{q}^\top) \breve{\boldsymbol{w}} + (\breve{\boldsymbol{\tau}} - 2\boldsymbol{q}\tau_N)^\top \breve{\boldsymbol{w}} - \frac{1}{2} \tau_N^2 \right\} \\ &= \max_{\gamma \in \mathbb{R}} \left\{ -\frac{1}{2}\gamma \; : \; \begin{array}{l} \breve{\boldsymbol{w}}^\top (\boldsymbol{Q} - 2\boldsymbol{q}\boldsymbol{q}^\top) \breve{\boldsymbol{w}} + (\breve{\boldsymbol{\tau}} - 2\boldsymbol{q}\tau_N)^\top \breve{\boldsymbol{w}} - \frac{1}{2}\tau_N^2 \geq -\frac{1}{2}\gamma, \\ \forall \breve{\boldsymbol{w}} \in \mathbb{R}^N \end{array} \right\} \\ &= \max_{\gamma \in \mathbb{R}} \left\{ -\frac{1}{2}\gamma \; : \; \begin{pmatrix} \boldsymbol{Q} - 2\boldsymbol{q}\boldsymbol{q}^\top & \frac{1}{2}(\breve{\boldsymbol{\tau}} - 2\boldsymbol{q}\tau_N) \\ \frac{1}{2}(\breve{\boldsymbol{\tau}} - 2\boldsymbol{q}\tau_N)^\top & \frac{1}{2}(\gamma - \tau_N^2) \end{pmatrix} \succeq 0 \right\}, \end{aligned} \tag{6.6}$$

where the second equality comes from minimizing the function with respect to w_N.

Using (6.6) instead of (4.4) for the function $H(\boldsymbol{\lambda}, \boldsymbol{\tau}; \boldsymbol{h})$ and again using the variable $\boldsymbol{r}$ in (6.1) leads to the following optimization problem that is equivalent to (RD):

$$\hat{\boldsymbol{r}} = \mathop{\arg\min}_{\boldsymbol{r} \in \mathbb{R}^{N(N+1)/2}} \breve{\mathcal{B}}_{\text{D1}}(\boldsymbol{r}, N, L, R), \tag{RD1}$$

where

$$\breve{\mathcal{B}}_{\mathrm{D1}}(\boldsymbol{r}, N, L, R) \triangleq \min_{\substack{\boldsymbol{\lambda}\in\mathbb{R}^N,\\ \boldsymbol{\tau}\in\mathbb{R}^{N+1},\\ \gamma\in\mathbb{R}}} \left\{\frac{1}{2}LR^2\gamma \;:\; \begin{pmatrix} \breve{\boldsymbol{Q}} - 2\breve{\boldsymbol{q}}\breve{\boldsymbol{q}}^\top & \frac{1}{2}(\breve{\boldsymbol{\tau}} - 2\breve{\boldsymbol{q}}\tau_N) \\ \frac{1}{2}(\breve{\boldsymbol{\tau}} - 2\breve{\boldsymbol{q}}\tau_N)^\top & \frac{1}{2}(\gamma - \tau_N^2) \end{pmatrix} \succeq 0,\ (\boldsymbol{\lambda}, \boldsymbol{\tau}) \in \Lambda \right\},$$

$$\begin{aligned}
\breve{\boldsymbol{Q}}(\boldsymbol{r}, \boldsymbol{\lambda}, \boldsymbol{\tau}) =& \frac{1}{2}\sum_{i=1}^{N-1} \lambda_i(\breve{\boldsymbol{u}}_{i-1} - \breve{\boldsymbol{u}}_i)(\breve{\boldsymbol{u}}_{i-1} - \breve{\boldsymbol{u}}_i)^\top + \frac{1}{2}\lambda_N\breve{\boldsymbol{u}}_{N-1}\breve{\boldsymbol{u}}_{N-1}^\top \\
&+ \frac{1}{2}\sum_{i=0}^{N-1}\tau_i\breve{\boldsymbol{u}}_i\breve{\boldsymbol{u}}_i^\top + \frac{1}{2}\sum_{i=1}^{N-1}\sum_{k=0}^{i-1} r_{i,k}(\breve{\boldsymbol{u}}_i\breve{\boldsymbol{u}}_k^\top + \breve{\boldsymbol{u}}_k\breve{\boldsymbol{u}}_i^\top), && (6.7) \\
\breve{\boldsymbol{q}}(\boldsymbol{r}, \boldsymbol{\lambda}, \boldsymbol{\tau}) =& \frac{1}{2}\sum_{k=0}^{N-2} r_{N,k}\breve{\boldsymbol{u}}_k, + \frac{1}{2}(r_{N,N-1} - \lambda_N)\breve{\boldsymbol{u}}_{N-1} && (6.8)
\end{aligned}$$

for $\breve{\boldsymbol{u}}_i = \boldsymbol{e}_{N,i+1} \in \mathbb{R}^N$. We omit the arguments $(\boldsymbol{r}, \boldsymbol{\lambda}, \boldsymbol{\tau})$ in $\breve{\boldsymbol{Q}}(\boldsymbol{r}, \boldsymbol{\lambda}, \boldsymbol{\tau})$ and $\breve{\boldsymbol{q}}(\boldsymbol{r}, \boldsymbol{\lambda}, \boldsymbol{\tau})$ for notational simplicity. Unlike (RD), we observe that the new equivalent form (RD1) has a point at the boundary of the positive semidefinite condition, and we will show that the point is indeed an optimal solution of both (RD) and (RD1).

Lemma 2 *A feasible point of both* (RD) *and* (RD1) *is* $(\hat{\boldsymbol{r}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}}, \hat{\gamma})$, *where*

$$\begin{aligned}
\hat{r}_{i,k} &= \begin{cases} \frac{4\theta_i\theta_k}{\theta_N^2}, & i = 2, \cdots, N-1,\ k = 0, \cdots, i-2, \\ \frac{4\theta_i\theta_{i-1}}{\theta_N^2} + \frac{2\theta_{i-1}^2}{\theta_N^2}, & i = 1, \cdots, N-1,\ k = i-1, \\ \frac{2\theta_k}{\theta_N}, & i = N,\ k = 0, \cdots, i-2, \\ \frac{2\theta_{N-1}}{\theta_N} + \frac{2\theta_{N-1}^2}{\theta_N^2}, & i = N,\ k = i-1, \end{cases} && (6.9) \\
\hat{\lambda}_i &= \frac{2\theta_{i-1}^2}{\theta_N^2}, \quad i = 1, \cdots, N, && (6.10) \\
\hat{\tau}_i &= \begin{cases} \frac{2\theta_i}{\theta_N^2}, & i = 0, \cdots, N-1, \\ 1 - \frac{2\theta_{N-1}^2}{\theta_N^2} = \frac{1}{\theta_N}, & i = N, \end{cases} && (6.11) \\
\hat{\gamma} &= \frac{1}{\theta_N^2}, && (6.12)
\end{aligned}$$

for

$$\theta_i = \begin{cases} 1, & i = 0, \\ \frac{1+\sqrt{1+4\theta_{i-1}^2}}{2}, & i = 1, \cdots, N-1, \\ \frac{1+\sqrt{1+8\theta_{i-1}^2}}{2} & i = N. \end{cases} \qquad (6.13)$$

Proof *The following set of conditions are sufficient for the feasible conditions of* (RD1)*:*

$$\begin{cases} \breve{\boldsymbol{Q}}(\boldsymbol{r},\boldsymbol{\lambda},\boldsymbol{\tau}) = 2\breve{\boldsymbol{q}}(\boldsymbol{r},\boldsymbol{\lambda},\boldsymbol{\tau})\breve{\boldsymbol{q}}(\boldsymbol{r},\boldsymbol{\lambda},\boldsymbol{\tau})^\top, \\ \breve{\boldsymbol{\tau}} = 2\breve{\boldsymbol{q}}(\boldsymbol{r},\boldsymbol{\lambda},\boldsymbol{\tau})\tau_N, \\ \gamma = \tau_N^2, \\ (\boldsymbol{\lambda},\boldsymbol{\tau}) \in \Lambda. \end{cases} \tag{6.14}$$

The Appendix shows that the point $(\hat{\boldsymbol{r}},\hat{\boldsymbol{\lambda}},\hat{\boldsymbol{\tau}},\hat{\gamma})$ *in* (6.9), (6.10), (6.11) *and* (6.12) *is the unique solution of* (6.14) *and also satisfies the feasible conditions of* (RD). □

Note that the parameter θ_i (6.13) used in Lemma 2 differs from t_i (3.1) only at the last iteration N. In other words, $\{\theta_0,\cdots,\theta_{N-1}\}$ is equivalent to $\{t_0,\cdots,t_{N-1}\}$ in (3.1) satisfying (3.2), whereas the last parameter θ_N satisfies

$$\theta_N^2 - \theta_N - 2\theta_{N-1}^2 = 0. \tag{6.15}$$

The next lemma shows that the feasible point derived in Lemma 2 is an optimal solution of both (RD) and (RD1).

Lemma 3 *The choice of* $(\hat{\boldsymbol{r}},\hat{\boldsymbol{\lambda}},\hat{\boldsymbol{\tau}},\hat{\gamma})$ *in* (6.9), (6.10), (6.11) *and* (6.12) *is an optimal solution of both* (RD) *and* (RD1).

Proof *A proof based on the Karush-Kuhn-Tucker (KKT) conditions is given in a previous version of this manuscript [7]. We removed this long proof (by reviewer request) since it is not necessary for the proof of Theorem 2, the main result of this paper, and to make room for the Discussion section.* □

The optimized step coefficients $\hat{\boldsymbol{h}}$ of interest are then derived using (6.3) with the analytical optimal solution $(\hat{\boldsymbol{r}},\hat{\boldsymbol{\lambda}},\hat{\boldsymbol{\tau}},\hat{\gamma})$ of (RD). It is interesting to note that the corresponding coefficients $\hat{\boldsymbol{h}}$ in (6.16) below have a recursive form that is similar to (3.4) of FGM2, as discussed further in Section 7.

Lemma 4 *The choice of* $(\hat{\boldsymbol{h}},\hat{\boldsymbol{\lambda}},\hat{\boldsymbol{\tau}},\hat{\gamma})$ *in* (6.10), (6.11), (6.12) *and*

$$\hat{h}_{i+1,k} = \begin{cases} \frac{1}{\theta_{i+1}}\left(2\theta_k - \sum_{j=k+1}^{i}\hat{h}_{j,k}\right), & k = 0,\cdots,i-1, \\ 1 + \frac{2\theta_i - 1}{\theta_{i+1}}, & k = i, \end{cases} \tag{6.16}$$

for $i = 0,\cdots,N-1$ *with* θ_i *in* (6.13) *is an optimal solution of* (HD).

Proof *Inserting* $\hat{\boldsymbol{r}}$ (6.9), $\hat{\boldsymbol{\lambda}}$ (6.10) *and* $\hat{\boldsymbol{\tau}}$ (6.11) *into* (6.3), *and noting that* $\hat{\lambda}_i + \hat{\tau}_i > 0$ *for* $i = 1,\cdots,N$, *we get*

$$\begin{aligned} \hat{h}_{i,k} &= \frac{\hat{r}_{i,k} - \hat{\tau}_i\sum_{j=k+1}^{i-1}\hat{h}_{j,k}}{\hat{\lambda}_i + \hat{\tau}_i}, \quad i = 1,\cdots,N,\ k = 0,\cdots,i-1, \\ &= \begin{cases} \frac{\theta_N^2}{2\theta_i^2}\left(\frac{4\theta_i\theta_k}{\theta_N^2} - \frac{2\theta_i}{\theta_N^2}\sum_{j=k+1}^{i-1}\hat{h}_{j,k}\right), & i = 1,\cdots,N-1,\ k = 0,\cdots,i-2, \\ \frac{\theta_N^2}{2\theta_i^2}\left(\frac{4\theta_i\theta_{i-1}}{\theta_N^2} + \frac{2\theta_{i-1}^2}{\theta_N^2}\right) = \frac{2\theta_i\theta_{i-1} + \theta_i^2 - \theta_i}{\theta_i^2}, & i = 1,\cdots,N-1,\ k = i-1, \\ \frac{2\theta_k}{\theta_N} - \frac{1}{\theta_N}\sum_{j=k+1}^{N-1}\hat{h}_{j,k}, & i = N,\ k = 0,\cdots,i-2, \\ \frac{2\theta_{N-1}}{\theta_N} + \frac{2\theta_{N-1}^2}{\theta_N^2} = \frac{2\theta_N\theta_{N-1} + \theta_N^2 - \theta_N}{\theta_N^2}, & i = N,\ k = i-1, \end{cases} \end{aligned}$$

which is equivalent to (6.16)*. From [5, Theorem 3], the corresponding* $(\hat{\boldsymbol{h}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}}, \hat{\gamma})$ *becomes an optimal solution of* (HD)*.* □

The following theorem shows that Algorithm FO with the optimized $\hat{\boldsymbol{h}}$ (6.16) achieves a new convergence bound.

Theorem 2 *Let* $f : \mathbb{R}^d \to \mathbb{R}$ *be convex and* $C_L^{1,1}$ *and let* $\boldsymbol{x}_0, \cdots, \boldsymbol{x}_N \in \mathbb{R}^d$ *be generated by Algorithm FO with* $\hat{\boldsymbol{h}}$ (6.16) *for a given* $N \geq 1$*. Then*

$$f(\boldsymbol{x}_N) - f(\boldsymbol{x}_*) \leq \frac{L||\boldsymbol{x}_0 - \boldsymbol{x}_*||^2}{2\theta_N^2} \leq \frac{L||\boldsymbol{x}_0 - \boldsymbol{x}_*||^2}{(N+1)(N+1+\sqrt{2})}, \quad \forall \boldsymbol{x}_* \in X_*(f). \tag{6.17}$$

Proof *Using* $\hat{\gamma}$ (6.12) *and* $\theta_{N-1}^2 = t_{N-1}^2 \geq \frac{(N+1)^2}{4}$ *from* (3.2) *and* (6.13)*, we get*

$$\begin{aligned}\hat{\gamma} = \frac{1}{\theta_N^2} &= \frac{4}{\left(1 + \sqrt{1 + 8\theta_{N-1}^2}\right)^2} \leq \frac{4}{\left(1 + \sqrt{1 + 2(N+1)^2}\right)^2} \\ &\leq \frac{2}{(N+1)^2 + \sqrt{2}(N+1) + 1} \leq \frac{2}{(N+1)(N+1+\sqrt{2})}.\end{aligned}$$

Then, we have

$$f(\boldsymbol{x}_N) - f(\boldsymbol{x}_*) \leq \mathcal{B}_{\mathrm{D}}(\hat{\boldsymbol{h}}, N, L, R) = \frac{1}{2}LR^2\hat{\gamma} \leq \frac{LR^2}{(N+1)(N+1+\sqrt{2})}, \quad \forall \boldsymbol{x}_* \in X_*(f),$$

based on Lemma 4. Finally, we let $R = ||\boldsymbol{x}_0 - \boldsymbol{x}_*||$*.* □

Theorem 2 shows that algorithm FO with the optimized $\hat{\boldsymbol{h}}$ (6.16) decreases the function f with a bound that is twice as small as that of Nesterov's fast gradient methods in (3.5) and (5.5), confirming DT's numerical results in [5, Tables 1 and 2]. The proposed algorithm requires at most $N = \left\lceil \sqrt{\frac{L}{\epsilon}} ||\boldsymbol{x}_0 - \boldsymbol{x}_*|| \right\rceil$ iterations to achieve the desired accuracy $f(\boldsymbol{x}_N) - f(\boldsymbol{x}_*) \leq \epsilon$, while Nesterov's fast gradient methods require at most $N = \left\lceil \sqrt{\frac{2L}{\epsilon}} ||\boldsymbol{x}_0 - \boldsymbol{x}_*|| \right\rceil$, a factor of about $\sqrt{2}$-times more iterations.

The next section describes efficient implementations of the corresponding Algorithm FO with $\hat{\boldsymbol{h}}$ (6.16).

7 Efficient formulations of proposed optimized first-order algorithms

Even though the analytical expression for $\hat{\boldsymbol{h}}$ in (6.16) that solves (HD) does not require an expensive SDP method, using $\hat{\boldsymbol{h}}$ in Algorithm FO would still be computationally undesirable. Noticing the similarity between (3.4) of FGM2 and (6.16), we can expect that Algorithm FO with (6.16) may have an equivalent efficient form as FGM2, as described next. In addition, we find an equivalent form of (6.16) that is similar to (3.3) of FGM1, so that we can find a formulation that is similar to FGM1 by analogy with how Proposition 2 shows the equivalence between (3.3) and (3.4).

Proposition 3 *The optimized $\hat{\boldsymbol{h}}$ in (6.16) satisfies the following recursive relationship*

$$\hat{h}_{i+1,k} = \begin{cases} \frac{\theta_i - 1}{\theta_{i+1}} \hat{h}_{i,k}, & k = 0, \cdots, i-2, \\ \frac{\theta_i - 1}{\theta_{i+1}} (\hat{h}_{i,i-1} - 1), & k = i-1, \\ 1 + \frac{2\theta_i - 1}{\theta_{i+1}}, & k = i, \end{cases} \tag{7.1}$$

for $i = 0, \cdots, N-1$ with θ_i in (6.13).

Proof *We follow the induction proof of Proposition 2 showing the equivalence between (3.3) and (3.4). We use the notation $\hat{h}'_{i,k}$ for the coefficient (6.16) to distinguish from (7.1).*

It is obvious that $\hat{h}'_{i+1,i} = \hat{h}_{i+1,i}$, $i = 0, \cdots, N-1$, and we clearly have

$$\begin{aligned} \hat{h}'_{i+1,i-1} &= \frac{1}{\theta_{i+1}} \left(2\theta_{i-1} - \hat{h}'_{i,i-1} \right) = \frac{1}{\theta_{i+1}} \left(2\theta_{i-1} - \left(1 + \frac{2\theta_{i-1} - 1}{\theta_i} \right) \right) \\ &= \frac{(2\theta_{i-1} - 1)(\theta_i - 1)}{\theta_i \theta_{i+1}} = \frac{\theta_i - 1}{\theta_{i+1}} \left(\hat{h}_{i,i-1} - 1 \right) = \hat{h}_{i+1,i-1}. \end{aligned}$$

for $i = 0, \cdots, N-1$.

We next use induction by assuming $\hat{h}'_{i+1,k} = \hat{h}_{i+1,k}$ for $i = 0, \cdots, n-1$, $k = 0, \cdots, i$. We then have

$$\begin{aligned} \hat{h}'_{n+1,k} &= \frac{1}{\theta_{n+1}} \left(2\theta_k - \sum_{j=k+1}^{n} \hat{h}'_{j,k} \right) = \frac{1}{\theta_{n+1}} \left(2\theta_k - \sum_{j=k+1}^{n-1} \hat{h}'_{j,k} - \hat{h}'_{n,k} \right) \\ &= \frac{\theta_n - 1}{\theta_{n+1}} \hat{h}'_{n,k} = \frac{\theta_n - 1}{\theta_{n+1}} \hat{h}_{n,k} = \hat{h}_{n+1,k} \end{aligned}$$

for $k = 1, \cdots, n-2$. Note that this proof is independent of the choice of θ_i. $\square$

Next, we revisit the derivation in Section 3 to transform Algorithm FO with (6.16) or (7.1) into efficient formulations akin to Nesterov's fast gradient methods, leading to practical algorithms.

7.1 Proposed optimized gradient method 1 (OGM1)

We first propose the following optimized gradient method, called OGM1, using (7.1) in Algorithm FO. OGM1 is computationally similar to FGM1 yet the sequence $\{\boldsymbol{x}_i\}$ generated by OGM1 achieves the fast convergence bound in Theorem 2.

```
Algorithm OGM1
  Input: $f \in C_L^{1,1}(\mathbb{R}^d)$ convex, $\boldsymbol{x}_0 \in \mathbb{R}^d$, $\boldsymbol{y}_0 = \boldsymbol{x}_0$, $\theta_0 = 1$.
  For $i = 0, \cdots, N-1$
      $\boldsymbol{y}_{i+1} = \boldsymbol{x}_i - \frac{1}{L} f'(\boldsymbol{x}_i)$
      $\theta_{i+1} = \begin{cases} \frac{1+\sqrt{1+4\theta_i^2}}{2}, & i \le N-2 \\ \frac{1+\sqrt{1+8\theta_i^2}}{2}, & i = N-1 \end{cases}$
      $\boldsymbol{x}_{i+1} = \boldsymbol{y}_{i+1} + \frac{\theta_i - 1}{\theta_{i+1}}(\boldsymbol{y}_{i+1} - \boldsymbol{y}_i) + \frac{\theta_i}{\theta_{i+1}}(\boldsymbol{y}_{i+1} - \boldsymbol{x}_i)$
```

Apparently, the proposed OGM1 accelerates FGM1 by using just one additional momentum term $\frac{\theta_i}{\theta_{i+1}}(\boldsymbol{y}_{i+1} - \boldsymbol{x}_i)$, and thus OGM1 is computationally efficient. Also, unlike DT's approach that requires choosing N for using a SDP solver before iterating, the proposed OGM1 does not need to know N in advance because the coefficients $\hat{\boldsymbol{h}}$ (or θ_i) for intermediate iterations ($i = 0, \cdots, N-1$) do not depend on N.

Proposition 4 *The sequence $\{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_N\}$ generated by Algorithm FO with (7.1) is identical to the corresponding sequence generated by Algorithm OGM1.*

Proof *We use induction, and for clarity, we use the notation $\boldsymbol{x}'_0, \cdots, \boldsymbol{x}'_N$ for Algorithm FO. It is obvious that $\boldsymbol{x}'_0 = \boldsymbol{x}_0$, and since $\theta_0 = 1$ we get*

$$\boldsymbol{x}'_1 = \boldsymbol{x}'_0 - \frac{1}{L}\hat{h}_{1,0} f'(\boldsymbol{x}'_0) = \boldsymbol{x}_0 - \frac{1}{L}\left(1 + \frac{2\theta_0 - 1}{\theta_1}\right) f'(\boldsymbol{x}_0) = \boldsymbol{y}_1 + \frac{\theta_0}{\theta_1}(\boldsymbol{y}_1 - \boldsymbol{x}_0) = \boldsymbol{x}_1.$$

Assuming $\boldsymbol{x}'_i = \boldsymbol{x}_i$ for $i = 0, \cdots, n$, we then have

$$\begin{aligned}
\boldsymbol{x}'_{n+1} =& \boldsymbol{x}'_n - \frac{1}{L}\hat{h}_{n+1,n} f'(\boldsymbol{x}'_n) - \frac{1}{L}\hat{h}_{n+1,n-1} f'(\boldsymbol{x}'_{n-1}) - \frac{1}{L}\sum_{k=0}^{n-2} \hat{h}_{n+1,k} f'(\boldsymbol{x}'_k) \\
=& \boldsymbol{x}_n - \frac{1}{L}\left(1 + \frac{2\theta_n - 1}{\theta_{n+1}}\right) f'(\boldsymbol{x}_n) \\
& - \frac{\theta_n - 1}{\theta_{n+1}}(\hat{h}_{n,n-1} - 1) f'(\boldsymbol{x}_{n-1}) - \frac{1}{L}\sum_{k=0}^{n-2} \frac{\theta_n - 1}{\theta_{n+1}} \hat{h}_{n,k} f'(\boldsymbol{x}_k) \\
=& \boldsymbol{x}_n - \frac{1}{L}\left(1 + \frac{\theta_n}{\theta_{n+1}}\right) f'(\boldsymbol{x}_n) \\
& + \frac{\theta_n - 1}{\theta_{n+1}}\left(-\frac{1}{L} f'(\boldsymbol{x}_n) + \frac{1}{L} f'(\boldsymbol{x}_{n-1}) - \frac{1}{L}\sum_{k=0}^{n-1} \hat{h}_{n,k} f'(\boldsymbol{x}_k)\right) \\
=& \boldsymbol{y}_{n+1} + \frac{\theta_n}{\theta_{n+1}}(\boldsymbol{y}_{n+1} - \boldsymbol{x}_n) + \frac{\theta_n - 1}{\theta_{n+1}}\left(-\frac{1}{L} f'(\boldsymbol{x}_n) + \frac{1}{L} f'(\boldsymbol{x}_{n-1}) + \boldsymbol{x}_n - \boldsymbol{x}_{n-1}\right)
\end{aligned}$$

$$=\boldsymbol{y}_{n+1}+\frac{\theta_n-1}{\theta_{n+1}}(\boldsymbol{y}_{n+1}-\boldsymbol{y}_n)+\frac{\theta_n}{\theta_{n+1}}(\boldsymbol{y}_{n+1}-\boldsymbol{x}_n)=\boldsymbol{x}_{n+1}$$

□

7.2 Proposed optimized gradient method 2 (OGM2)

We propose another efficient formulation of Algorithm FO with (6.16) that is similar to the formulation of FGM2.

```
Algorithm OGM2
  Input: $f \in C_L^{1,1}(\mathbb{R}^d)$ convex, $\boldsymbol{x}_0 \in \mathbb{R}^d$, $\theta_0 = 1$.
  For $i = 0, \cdots, N-1$
      $\boldsymbol{y}_{i+1} = \boldsymbol{x}_i - \frac{1}{L} f'(\boldsymbol{x}_i)$
      $\boldsymbol{z}_{i+1} = \boldsymbol{x}_0 - \frac{1}{L} \sum_{k=0}^{i} 2\theta_k f'(\boldsymbol{x}_k)$
      $\theta_{i+1} = \begin{cases} \frac{1+\sqrt{1+4\theta_i^2}}{2}, & i \le N-2 \\ \frac{1+\sqrt{1+8\theta_i^2}}{2}, & i = N-1 \end{cases}$
      $\boldsymbol{x}_{i+1} = \left(1 - \frac{1}{\theta_{i+1}}\right) \boldsymbol{y}_{i+1} + \frac{1}{\theta_{i+1}} \boldsymbol{z}_{i+1}$
```

The sequence $\{\boldsymbol{x}_i\}$ generated by OGM2 achieves the fast convergence bound in Theorem 2. Algorithm OGM2 doubles the weight on all previous gradients for $\{\boldsymbol{z}_i\}$ compared to FGM2, providing some intuition for its two-fold acceleration. OGM2 requires comparable computation per iteration as FGM2.

Proposition 5 *The sequence $\{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_N\}$ generated by Algorithm FO with (6.16) is identical to the corresponding sequence generated by Algorithm OGM2.*

Proof *We use induction, and for clarity, we use the notation $\boldsymbol{x}'_0, \cdots, \boldsymbol{x}'_N$ for Algorithm FO. It is obvious that $\boldsymbol{x}'_0 = \boldsymbol{x}_0$, and since $\theta_0 = 1$ we get*

$$\boldsymbol{x}'_1=\boldsymbol{x}'_0-\frac{1}{L}\hat{h}_{1,0}f'(\boldsymbol{x}'_0)=\boldsymbol{x}_0-\frac{1}{L}\left(1+\frac{2\theta_0-1}{\theta_1}\right)f'(\boldsymbol{x}_0)=\boldsymbol{y}_1+\frac{\theta_0}{\theta_1}(\boldsymbol{y}_1-\boldsymbol{x}_0)=\boldsymbol{x}_1.$$

Assuming $\boldsymbol{x}'_i = \boldsymbol{x}_i$ for $i = 0, \cdots, n$, we then have

$$\begin{aligned}\boldsymbol{x}'_{n+1}=&\boldsymbol{x}'_n-\frac{1}{L}\hat{h}_{n+1,n}f'(\boldsymbol{x}'_n)-\frac{1}{L}\sum_{k=0}^{n-1}\hat{h}_{n+1,k}f'(\boldsymbol{x}'_k)\\ =&\boldsymbol{x}_n-\frac{1}{L}\left(1+\frac{2\theta_n-1}{\theta_{n+1}}\right)f'(\boldsymbol{x}_n)-\frac{1}{L}\sum_{k=0}^{n-1}\frac{1}{\theta_{n+1}}\left(2\theta_k-\sum_{j=k+1}^{n}\hat{h}_{j,k}\right)f'(\boldsymbol{x}_k)\end{aligned}$$

$$= \left(1 - \frac{1}{\theta_{n+1}}\right)\left(\boldsymbol{x}_n - \frac{1}{L} f'(\boldsymbol{x}_n)\right) + \frac{1}{\theta_{n+1}}\left(\boldsymbol{x}_0 - \frac{1}{L}\sum_{k=0}^{n} 2\theta_k f'(\boldsymbol{x}_k)\right)$$
$$= \left(1 - \frac{1}{\theta_{n+1}}\right)\boldsymbol{y}_{n+1} + \frac{1}{\theta_{n+1}}\boldsymbol{z}_{n+1} = \boldsymbol{x}_{n+1}$$

The third equality uses the telescoping sum $\boldsymbol{x}_n = \boldsymbol{x}_0 + \sum_{j=1}^{n}(\boldsymbol{x}_j - \boldsymbol{x}_{j-1})$ and (1.1) in Algorithm FO. □

8 Discussion

After submitting this work [7], Taylor *et al.* [17] further studied the PEP approach to compute the exact worst-case bound of first-order methods, unlike DT [5] and this paper that use the *relaxed* PEP. Taylor *et al.* [17] studied the tightness of relaxations on PEP introduced in [5] and avoided some strict relaxations.

Inspired by [17, Conjecture 5], we developed the following theorem that shows that the smallest upper bound in (6.17) for OGM1 and OGM2 is tight, despite the various relaxations of PEP used in [5] and herein. (Similar tightness results are shown for the gradient methods with a constant step size $0 < h \le 1$ in [5].) The following theorem specifies a worst-case convex function $\phi(\boldsymbol{x})$ in $C_L^{1,1}(\mathbb{R}^d)$ for which the optimized gradient methods achieve their smallest upper bound in (6.17).

Theorem 3 *For the following convex functions in $C_L^{1,1}(\mathbb{R}^d)$ for all $d \ge 1$:*

$$\phi(\boldsymbol{x}) = \begin{cases} \frac{LR}{\theta_N^2}||\boldsymbol{x}|| - \frac{LR^2}{2\theta_N^4}, & \text{if } ||\boldsymbol{x}|| \ge \frac{R}{\theta_N^2}, \\ \frac{L}{2}||\boldsymbol{x}||^2, & \text{otherwise} \end{cases} \tag{8.1}$$

both OGM1 and OGM2 exactly achieve the smallest upper bound in (6.17), i.e.,

$$\phi(\boldsymbol{x}_N) - \phi(\boldsymbol{x}_*) = \frac{L||\boldsymbol{x}_0 - \boldsymbol{x}_*||^2}{2\theta_N^2}.$$

Proof We show in the Appendix that the following property of the coefficients $\hat{\boldsymbol{h}}$ (7.1) of OGM1 and OGM2 holds:

$$\sum_{j=1}^{i}\sum_{k=0}^{j-1}\hat{h}_{j,k} = \begin{cases} \theta_i^2 - 1, & i = 1, \cdots, N-1, \\ \frac{1}{2}(\theta_N^2 - 1), & i = N, \end{cases} \tag{8.2}$$

Then, starting from $\boldsymbol{x}_0 = R\boldsymbol{\nu}$, where $\boldsymbol{\nu}$ is a unit vector, and using (8.2), the iterates of OGM1 and OGM2 are as follows

$$\boldsymbol{x}_i = \boldsymbol{x}_0 - \frac{1}{L}\sum_{j=1}^{i}\sum_{k=0}^{j-1}\hat{h}_{j,k}\phi'(\boldsymbol{x}_k) = \begin{cases} \left(1 - \frac{\theta_i^2 - 1}{\theta_N^2}\right)R\boldsymbol{\nu}, & i = 0, \cdots, N-1, \\ \left(1 - \frac{\theta_N^2 - 1}{2\theta_N^2}\right)R\boldsymbol{\nu}, & i = N, \end{cases}$$

where the corresponding sequence $\{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_N\}$ stays in the affine region of the function $\phi(\boldsymbol{x})$ with the same gradient value:

$$\phi'(\boldsymbol{x}_i) = \frac{LR}{\theta_N^2}\boldsymbol{\nu}, \quad i = 0, \cdots, N.$$

Therefore, after N iterations of OGM1 and OGM2, we have

$$\phi(\boldsymbol{x}_N) - \phi(\boldsymbol{x}_*) = \phi(\boldsymbol{x}_N) = \frac{LR^2}{2\theta_N^2},$$

exactly matching the smallest upper bound in (6.17). □

This result implies that the exact PEP bound $\mathcal{B}_{\text{P}}(\hat{\boldsymbol{h}}, N, d, L, R)$ of OGM1 and OGM2 is equivalent to their relaxed bound $\mathcal{B}_{\text{D}}(\hat{\boldsymbol{h}}, N, L, R)$ that is independent of d. Note that Taylor *et al.* [17] showed that the exact PEP $\mathcal{B}_{\text{P}}(\boldsymbol{h}, N, d, L, R)$ is independent of d. Whereas the OGM bound (6.17) is tight, the FGM bounds (3.5) and (5.5) are not tight [5,17], somewhat weakening the utility of the fact that the OGM bound (6.17) is twice smaller than the FGM bounds. However, Figure 5 in [17] shows that the FGM bounds (3.5) and (5.5) become close to tight asymptotically as N increases, so the factor of 2 can have practical value when using many iterations. We leave more complete comparisons as future work.

9 Conclusion

We proposed new optimized first-order algorithms that achieve a worst-case convergence bound that is twice as small as that of Nesterov's methods for smooth unconstrained convex minimization, inspired by [5]. The proposed first-order methods are comparably efficient for implementation as Nesterov's methods. Thus it is natural to use the proposed OGM1 and OGM2 to replace Nesterov's methods in smooth unconstrained convex minimization. Numerical results in large-scale imaging applications show practical convergence acceleration consistent with those predicted by the bounds given here [8,9]. Those applications use regularizers that have shapes somewhat similar to the worst-case function (8.1).

The efficient formulations of both Nesterov's methods and the new optimized first-order methods still seem somewhat magical. Recently, [1], [14] and [16] studied Nesterov's FGM formulations, and extending such studies to the new OGM methods should further illuminate the fundamental causes for their efficient formulations and acceleration. Also, new optimized first-order methods lack analytical convergence bounds for the intermediate iterations, whereas numerical bounds are studied in [17]; deriving those analytical bounds is interesting future work.

Drori recently extended the PEP approach to projected gradient methods for constrained smooth convex minimization [4]. Extending this approach to general first-order algorithms including our proposed OGM1 and OGM2 is important future work. In addition, just as Nesterov's fast gradient methods have been extended for nonsmooth composite convex minimization [2,13], extending the proposed optimized first-order algorithms for minimizing nonsmooth composite convex functions would be a natural direction to pursue.

While DT's PEP approach involves a series of relaxations to make the problem solvable, OGM1 and OGM2 with the step coefficients $\hat{\boldsymbol{h}}$ that are optimized over the *relaxed* PEP upper bound (HD) achieve an *exact* bound in Theorem 3. However, it remains an open problem to either prove that the smallest upper bound in (6.17) of OGM1 and OGM2 is optimal. We leave either proving the above statement for (HP) or to further optimize the first-order methods as future work.

10 Appendix

10.1 Proof of Lemma 2

We prove that the choice $(\hat{\boldsymbol{r}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}}, \gamma)$ in (6.9), (6.10), (6.11) and (6.12) satisfies the feasible conditions (6.14) of (RD1).

Using the definition of $\breve{\boldsymbol{Q}}(\boldsymbol{r}, \boldsymbol{\lambda}, \boldsymbol{\tau})$ in (6.7), and considering the first two conditions of (6.14), we get

$$\begin{aligned}\lambda_{i+1} = \breve{Q}_{i,i}(\boldsymbol{r}, \boldsymbol{\lambda}, \boldsymbol{\tau}) = 2\breve{q}_i^2(\boldsymbol{r}, \boldsymbol{\lambda}, \boldsymbol{\tau}) &= \frac{1}{2\tau_N^2}\tau_i^2 \\ &= \begin{cases} \frac{1}{2(1-\lambda_N)^2}\lambda_1^2, & i = 0 \\ \frac{1}{2(1-\lambda_N)^2}(\lambda_{i+1} - \lambda_i)^2, & i = 1, \cdots, N-1, \end{cases}\end{aligned}$$

where the last equality comes from $(\boldsymbol{\lambda}, \boldsymbol{\tau}) \in \Lambda$, and this reduces to the following recursion:

$$\begin{cases} \lambda_1 = 2(1-\lambda_N)^2, \\ (\lambda_i - \lambda_{i-1})^2 - \lambda_1\lambda_i = 0. \quad i = 2, \cdots, N. \end{cases} \tag{10.1}$$

We use induction to prove that the solution of (10.1) is

$$\lambda_i = \begin{cases} \frac{2}{\theta_N^2}, & i = 1, \\ \theta_{i-1}^2\lambda_1, & i = 2, \cdots, N, \end{cases}$$

which is equivalent to $\hat{\boldsymbol{\lambda}}$ (6.10). It is obvious that $\lambda_1 = \theta_0\lambda_1$, and for $i = 2$ in (10.1), we get

$$\lambda_2 = \frac{3\lambda_1 + \sqrt{9\lambda_1^2 - 4\lambda_1^2}}{2} = \frac{3+\sqrt{5}}{2}\lambda_1 = \theta_1^2\lambda_1.$$

Then, assuming $\lambda_i = \theta_{i-1}^2 \lambda_1$ for $i = 1, \cdots, n$ and $n \le N-1$, and using the second equality in (10.1) for $i = n+1$, we get

$$\begin{aligned}\lambda_{n+1} &= \frac{\lambda_1 + 2\lambda_n + \sqrt{(\lambda_1 + 2\lambda_n)^2 - 4\lambda_n^2}}{2} = \frac{1 + 2\theta_{n-1}^2 + \sqrt{1 + 4\theta_{n-1}^2}}{2}\lambda_1 \\ &= \left(\theta_{n-1}^2 + \frac{1 + \sqrt{1 + 4\theta_{n-1}^2}}{2}\right)\lambda_1 = \theta_n^2 \lambda_1,\end{aligned}$$

where the last equality uses (3.2). Then we use the first equality in (10.1) to find the value of λ_1 as

$$\begin{aligned}&\lambda_1 = 2(1 - \theta_{N-1}^2 \lambda_1)^2 \\ &\theta_{N-1}^4 \lambda_1^2 - 2\left(\theta_{N-1}^2 + \frac{1}{4}\right)\lambda_1 + 1 = 0 \\ &\lambda_1 = \frac{\theta_{N-1}^2 + \frac{1}{4} - \sqrt{(\theta_{N-1}^2 + \frac{1}{4})^2 - \theta_{N-1}^4}}{\theta_{N-1}^4} = \frac{1}{\theta_{N-1}^2 + \frac{1}{4} + \sqrt{\frac{\theta_{N-1}^2}{2} + \frac{1}{16}}} \\ &\quad = \frac{8}{\left(1 + \sqrt{1 + 8\theta_{N-1}^2}\right)^2} = \frac{2}{\theta_N^2}\end{aligned}$$

with θ_N in (6.13).

Until now, we derived $\hat{\boldsymbol{\lambda}}$ (6.10) using some conditions of (6.14). Consequently, using the last two conditions in (6.14) with (3.2) and (6.15), we can easily derive the following:

$$\begin{aligned}\tau_i &= \begin{cases}\hat{\lambda}_1 = \frac{2}{\theta_N^2}, & i = 0, \\ \hat{\lambda}_{i+1} - \hat{\lambda}_i = \frac{2\theta_i^2}{\theta_N^2} - \frac{2\theta_{i-1}^2}{\theta_N^2} = \frac{2\theta_i}{\theta_N^2}, & i = 1, \cdots, N-1, \\ 1 - \hat{\lambda}_N = 1 - \frac{2\theta_{N-1}^2}{\theta_N^2} = \frac{1}{\theta_N}, & i = N,\end{cases} \\ \gamma &= \tau_N^2 = \frac{1}{\theta_N^2},\end{aligned}$$

which are equivalent to $\hat{\boldsymbol{\tau}}$ (6.11) and $\hat{\gamma}$ (6.12).

Next, we derive $\hat{\boldsymbol{r}}$ for given $\hat{\boldsymbol{\lambda}}$ (6.10) and $\hat{\boldsymbol{\tau}}$ (6.11). Inserting $\hat{\boldsymbol{\tau}}$ (6.11) to the first two conditions of (6.14), we get

$$\begin{cases}\breve{q}_i(\hat{\boldsymbol{r}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}}) = \frac{\hat{\tau}_i}{2\hat{\tau}_N} = \frac{\theta_i}{\theta_N}, \\ \breve{Q}_{i,k}(\hat{\boldsymbol{r}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}}) = 2\breve{q}_i(\boldsymbol{r}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}})\breve{q}_k(\boldsymbol{r}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}}) = \frac{2\theta_i\theta_k}{\theta_N^2},\end{cases} \tag{10.2}$$

for $i, k = 0, \cdots, N-1$, and considering (6.5) and (10.2), we get

$$\breve{S}_{i,k}(\hat{\boldsymbol{r}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}}) = \begin{cases} \frac{2\theta_i\theta_k}{\theta_N^2}, & i,k = 0,\cdots,N-1, \\ \frac{\theta_i}{\theta_N} & i = 0,\cdots,N-1,\ k = N, \\ \frac{\theta_k}{\theta_N}, & i = N,\ k = 0,\cdots,N-1, \\ \frac{1}{2} & i = N,\ k = N. \end{cases} \tag{10.3}$$

Finally, using the two equivalent forms (6.2) and (10.3) of $\breve{\boldsymbol{S}}(\hat{\boldsymbol{r}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}})$, we get

$$\breve{S}_{i,k}(\hat{\boldsymbol{r}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}}) = \begin{cases} \frac{1}{2}\hat{r}_{i,k} = \frac{2\theta_i\theta_k}{\theta_N^2}, & i = 2,\cdots,N-1,\ k = 0,\cdots,i-2, \\ \frac{1}{2}(\hat{r}_{i,k} - \hat{\lambda}_i) = \frac{2\theta_i\theta_k}{\theta_N^2}, & i = 1,\cdots,N-1,\ k = i-1, \\ \frac{1}{2}\hat{r}_{i,k} = \frac{\theta_k}{\theta_N}, & i = N,\ k = 0,\cdots,i-2, \\ \frac{1}{2}(\hat{r}_{i,k} - \hat{\lambda}_i) = \frac{\theta_k}{\theta_N}. & i = N,\ k = i-1, \end{cases} \tag{10.4}$$

and this can be easily converted to the choice $\hat{r}_{i,k}$ in (6.9).

For these given $(\hat{\boldsymbol{r}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}})$, we can easily notice that

$$\begin{pmatrix} \breve{\boldsymbol{S}}(\hat{\boldsymbol{r}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\tau}}) & \frac{1}{2}\hat{\boldsymbol{\tau}} \\ \frac{1}{2}\hat{\boldsymbol{\tau}}^\top & \frac{1}{2}\hat{\gamma} \end{pmatrix} = \begin{pmatrix} \frac{2}{\theta_N^2}\breve{\boldsymbol{\theta}}\breve{\boldsymbol{\theta}}^\top & \frac{1}{\theta_N^2}\breve{\boldsymbol{\theta}} \\ \frac{1}{\theta_N^2}\breve{\boldsymbol{\theta}}^\top & \frac{1}{2\theta_N^2} \end{pmatrix} = \frac{2}{\theta_N^2}\begin{pmatrix} \breve{\boldsymbol{\theta}} \\ \frac{1}{2} \end{pmatrix}\begin{pmatrix} \breve{\boldsymbol{\theta}} \\ \frac{1}{2} \end{pmatrix}^\top \succeq 0 \tag{10.5}$$

for $\breve{\boldsymbol{\theta}} = \left(\theta_0, \cdots, \theta_{N-1}, \frac{\theta_N}{2}\right)^\top$, showing that the choice is feasible in both (RD) and (RD1). $\square$

10.2 Proof of (8.2)

We prove that (8.2) holds for the coefficients $\hat{\boldsymbol{h}}$ (7.1) of OGM1 and OGM2.

We first show the following property using induction:

$$\sum_{k=0}^{j-1} \hat{h}_{j,k} = \begin{cases} \theta_j, & j = 1,\cdots,N-1, \\ \frac{1}{2}(\theta_N + 1), & j = N. \end{cases}$$

Clearly, $\hat{h}_{1,0} = 1 + \frac{2\theta_0 - 1}{\theta_1} = \theta_1$ using (3.2). Assuming $\sum_{k=0}^{j-1} \hat{h}_{j,k} = \theta_j$ for $j = 1, \cdots, n$ and $n \le N-1$, we get

$$\begin{aligned} \sum_{k=0}^{n} \hat{h}_{n+1,k} &= 1 + \frac{2\theta_n - 1}{\theta_{n+1}} + \frac{\theta_n - 1}{\theta_{n+1}}(\hat{h}_{n,n-1} - 1) + \frac{\theta_n - 1}{\theta_{n+1}}\sum_{k=0}^{n-2} \hat{h}_{n,k} \\ &= 1 + \frac{\theta_n}{\theta_{n+1}} + \frac{\theta_n - 1}{\theta_{n+1}}\sum_{k=0}^{n-1} \hat{h}_{n,k} = \frac{\theta_{n+1} + \theta_n^2}{\theta_{n+1}} \\ &= \begin{cases} \theta_n, & n = 1,\cdots,N-2, \\ \frac{1}{2}(\theta_N + 1), & n = N-1, \end{cases} \end{aligned}$$

where the last equality uses (3.2) and (6.15).

Then, (8.2) can be easily derived using (3.2) and (6.15) as

$$\sum_{j=1}^{i}\sum_{k=0}^{j-1}\hat{h}_{j,k} = \begin{cases} \sum_{j=1}^{i}\theta_j, & i = 1,\cdots,N-1, \\ \sum_{j=1}^{N-1}\theta_j + \frac{1}{2}(\theta_N+1), & i = N, \end{cases}$$
$$= \begin{cases} \theta_i^2 - 1, & i = 1,\cdots,N-1, \\ \frac{1}{2}(\theta_N^2-1), & i = N. \end{cases}$$

□

References

1. Allen-Zhu, Z., Orecchia, L.: Linear coupling: An ultimate unification of gradient and mirror descent (2015). URL http://arxiv.org/abs/1407.1537. Arxiv 1407.1537
2. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imag. Sci. **2**(1), 183–202 (2009). DOI 10.1137/080716542
3. CVX Research, I.: CVX: Matlab software for disciplined convex programming, version 2.0. http://cvxr.com/cvx (2012)
4. Drori, Y.: Contributions to the complexity analysis of optimization algorithms. Ph.D. thesis, Tel-Aviv Univ., Israel (2014)
5. Drori, Y., Teboulle, M.: Performance of first-order methods for smooth convex minimization: A novel approach. Math. Program. **145**(1-2), 451–82 (2014). DOI 10.1007/s10107-013-0653-0
6. Grant, M., Boyd, S.: Graph implementations for nonsmooth convex programs. In: V. Blondel, S. Boyd, H. Kimura (eds.) Recent Advances in Learning and Control, Lecture Notes in Control and Information Sciences, pp. 95–110. Springer-Verlag Limited (2008). http://stanford.edu/~boyd/graph_dcp.html
7. Kim, D., Fessler, J.A.: Optimized first-order methods for smooth convex minimization (2014). URL http://arxiv.org/abs/1406.5468. Arxiv 1406.5468
8. Kim, D., Fessler, J.A.: Optimized momentum steps for accelerating X-ray CT ordered subsets image reconstruction. In: Proc. 3rd Intl. Mtg. on Imag. Form. in X-ray CT, pp. 103–6 (2014)
9. Kim, D., Fessler, J.A.: An optimized first-order method for image restoration. In: Proc. IEEE Intl. Conf. on Imag. Proc. (2015). To appear.
10. Nesterov, Y.: A method of solving a convex programming problem with convergence rate $O(1/k^2)$. Sov. Math. Dokl. **27**(2), 372–76 (1983)
11. Nesterov, Y.: Introductory lectures on convex optimization: A basic course. Kluwer (2004)
12. Nesterov, Y.: Smooth minimization of non-smooth functions. Math. Program. **103**(1), 127–52 (2005). DOI 10.1007/s10107-004-0552-5
13. Nesterov, Y.: Gradient methods for minimizing composite functions. Mathematical Programming **140**(1), 125–61 (2013). DOI 10.1007/s10107-012-0629-5
14. O'Donoghue, B., Candès, E.: Adaptive restart for accelerated gradient schemes. Found. Computational Math. **15**(3), 715–32 (2015). DOI 10.1007/s10208-013-9150-3
15. Polyak, B.T.: Some methods of speeding up the convergence of iteration methods. USSR Comp. Math. Math. Phys. **4**(5), 1–17 (1964). DOI 10.1016/0041-5553(64)90137-5
16. Su, W., Boyd, S., Candes, E.J.: A differential equation for modeling Nesterov's accelerated gradient method: theory and insights (2015). URL http://arxiv.org/abs/1503.01243. Arxiv 1503.01243
17. Taylor, A.B., Hendrickx, J.M., Glineur, François.: Smooth strongly convex interpolation and exact worst-case performance of first- order methods (2015). URL http://arxiv.org/abs/1502.05666. Arxiv 1502.05666
18. Tseng, P.: Approximation accuracy, gradient methods, and error bound for structured convex optimization. Math. Program. **125**(2), 263–95 (2010). DOI 10.1007/s10107-010-0394-2