

GRADIENT METHODS FOR STOCHASTIC OPTIMIZATION IN RELATIVE SCALE

Yurii Nesterov, Anton Rodomanov

LIDAM Discussion Paper CORE
2024 / 27

CORE

Voie du Roman Pays 34, L1.03.01

B-1348 Louvain-la-Neuve

Tel (32 10) 47 43 04

Email: immaq-library@uclouvain.be

<https://uclouvain.be/en/research-institutes/lidam/core/core-discussion-papers.html>

Gradient Methods for Stochastic Optimization in Relative Scale*

Yurii Nesterov[†]Anton Rodomanov[‡]

May 28, 2023

[version 0.3.0]

Abstract

We propose a new concept of a relatively inexact stochastic subgradient and present novel first-order methods that can use such objects to approximately solve convex optimization problems in relative scale. An important example where relatively inexact subgradients naturally arise is given by the Power or Lanczos algorithms for computing an approximate leading eigenvector of a symmetric positive semidefinite matrix. Using these algorithms as subroutines in our methods, we get new optimization schemes that can provably solve certain large-scale Semidefinite Programming problems with relative accuracy guarantees by using only matrix-vector products.

Keywords: convex optimization, optimization in relative scale, gradient methods, randomization, convergence guarantees, eigenvalues, singular values, power method, Lanczos algorithm

1 Introduction

1.1 Motivation

Semidefinite Programming (SDP) is an important class of optimization problems. The standard methods for solving SDP problems are Interior-Point Methods (IPMs) [6]. These methods are based on Newton steps and are very efficient for small- and medium-size problems. In many cases, IPMs are able to find an approximate solution with a high accuracy in several dozen of iterations. However, IPMs have a significant drawback: they cannot be used for large-scale problems that often arise in modern applications and for which computing even one Newton step becomes too expensive.

The only way to solve large-scale SDP problems is to use first-order methods relying on matrix-vector products. Compared to IPMs, these methods have much cheaper iterations

*This paper has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 788368).

[†]Center for Operations Research and Econometrics (CORE), Catholic University of Louvain (UCL). E-mail: yurii.nesterov@uclouvain.be.

[‡]Institute of Information and Communication Technologies, Electronics and Applied Mathematics (ICTEAM), Catholic University of Louvain (UCL). E-mail: anton.rodomanov@uclouvain.be.

and compute less accurate solutions. However, the accuracy is usually not a problem since, in the majority of applications involving large-scale problems, there is no need for high accuracy.

In this paper, we develop new algorithms for solving optimization problems in *relative scale* with provable efficiency guarantees. In contrast to already existing approaches, our methods can work with inexact (possibly stochastic) information about the objective function.

1.2 Notation and Generalities

In what follows, we denote by $\mathbb{E}$ a finite-dimensional real vector space, and by $\mathbb{E}^*$ its dual space, formed by all linear functions on $\mathbb{E}$. The value of function $s \in \mathbb{E}^*$ at point $x \in \mathbb{E}$ is denoted by $\langle s, x \rangle$.

Given a self-adjoint positive semidefinite linear operator $B: \mathbb{E} \rightarrow \mathbb{E}^*$, we can define the following Euclidean seminorm in $\mathbb{E}$:

$$\|x\|_B := \langle Bx, x \rangle^{1/2}, \quad x \in \mathbb{E}. \quad (1.1)$$

An important subspace for the seminorm $\|\cdot\|_B$ is the kernel $\ker B$. For any $x \in \mathbb{E}$, it holds that $\|x\|_B = 0$ iff $x \in \ker B$. Hence, the seminorm $\|\cdot\|_B$ is a norm iff B is nondegenerate. More generally, for (any) complementary subspace $(\ker B)^c \subseteq \mathbb{E}$ to $\ker B$, the restriction of $\|\cdot\|_B$ onto $(\ker B)^c$ is a norm. Note that $\|\cdot\|_B$ is constant along $\ker B$:

$$\|x + h\|_B = \|x\|_B, \quad \forall x \in \mathbb{E}, \forall h \in \ker B. \quad (1.2)$$

Each seminorm $\|\cdot\|_B$ induces the following (generalized) dual norm in $\mathbb{E}^*$:

$$\|s\|_B^* := \sup_{x \in \mathbb{E}} \{\langle s, x \rangle : \|x\|_B \leq 1\}, \quad s \in \mathbb{E}^*. \quad (1.3)$$

Strictly speaking, $\|\cdot\|_B^*$ is not a norm in $\mathbb{E}^*$, as it can take infinite values at certain points:¹

$$\|s\|_B^* < +\infty \iff s \in (\ker B)^\perp, \quad \forall s \in \mathbb{E}^*. \quad (1.4)$$

Nevertheless, when restricted to $(\ker B)^\perp$, $\|\cdot\|_B^*$ is indeed a norm. In the special case when B is nondegenerate, $(\ker B)^\perp = \mathbb{E}^*$ and $\|\cdot\|_B^*$ becomes a norm in $\mathbb{E}^*$ given by $\|s\|_B^* = \langle s, B^{-1}s \rangle^{1/2}$ for all $s \in \mathbb{E}^*$. The following identity is often useful:

$$\frac{1}{2}(\|s\|_B^*)^2 = \sup_{x \in \mathbb{E}} \left[\langle s, x \rangle - \frac{1}{2}\|x\|_B^2 \right], \quad \forall s \in \mathbb{E}^*. \quad (1.5)$$

If $\mathbb{E} = \mathbb{R}^n$, the space of n -dimensional real column vectors, then we often use the standard scalar product

$$\langle x, y \rangle = x^T y = \sum_{i=1}^n x^{(i)} y^{(i)}, \quad x, y \in \mathbb{R}^n.$$

For $x \in \mathbb{R}^n$, the standard Euclidean norm is defined as $\|x\| = \langle x, x \rangle^{1/2}$.

¹Hereinafter, for a linear subspace $\mathbb{L} \subseteq \mathbb{E}$, $\mathbb{L}^\perp := \{s \in \mathbb{E}^* : \langle s, x \rangle = 0, \forall x \in \mathbb{L}\}$ denotes the orthogonal complement of $\mathbb{L}$ in $\mathbb{E}$.

The standard Euclidean sphere in $\mathbb{R}^n$ is denoted by $\mathcal{S}^{n-1}$. The cone of n -dimensional vectors with nonnegative coordinates is denoted by $\mathbb{R}_+^n$. If all the coordinates are strictly positive, we use the notation $\mathbb{R}_{++}^n$.

The notation $\mathbb{R}^{m \times n}$ is used for the space of real $m \times n$ matrices equipped with the standard Frobenius inner product:

$$\langle X, Y \rangle = \sum_{i=1}^n \sum_{j=1}^m X^{(i,j)} Y^{(i,j)}, \quad X, Y \in \mathbb{R}^{m \times n}.$$

For $X \in \mathbb{R}^{m \times n}$, its standard Frobenius norm is defined as $\|X\|_F = \langle X, X \rangle^{1/2}$.

For the space of symmetric $n \times n$ matrices, we use notation $\mathbb{S}^n$. All eigenvalues of a matrix $X \in \mathbb{S}^n$ are real, and we denote by $\lambda_{\max}(X)$ and $\lambda_{\min}(X)$ the maximal and the minimal ones, respectively. A matrix $X \in \mathbb{S}^n$ is positive semidefinite iff $\lambda_{\min}(X) \geq 0$ (notation $X \succeq 0$). The convex cone of $n \times n$ real positive semidefinite matrices is denoted by $\mathbb{S}_+^n$.

For each matrix $A \in \mathbb{R}^{m \times n}$, we can construct two “squares” of A —the matrices $AA^T \in \mathbb{S}_+^m$ and $A^T A \in \mathbb{S}_+^n$ —that share $r := \min\{m, n\}$ common nonnegative eigenvalues. The square roots of these common eigenvalues—the numbers $\sigma_1, \dots, \sigma_r \in \mathbb{R}_+$ —are called the *singular values* of A . The largest of them is the *maximal singular value* of A denoted by $\sigma_{\max}(A)$.

Singular values are often used for defining matrix norms. For a matrix $A \in \mathbb{R}^{m \times n}$ and a real $p \in [0, \infty]$, the *Schatten p -norm* of A , denoted by $\|A\|_p$, is defined as the ℓ_p -norm of the vector of singular values of A . An important example is the Schatten infinity-norm also known as the *spectral norm*:

$$\|A\|_\infty := \sigma_{\max}(A) = \max_{\|x\|=1} \|Ax\|.$$

2 Optimization in Relative Scale

2.1 Gradient Method with Relatively Inexact Stochastic Oracle

Consider the following optimization problem:

$$f^* := \min_{x \in Q} f(x), \tag{2.1}$$

where $f: \mathbb{E} \rightarrow \mathbb{R}$ is a convex function and $Q \subseteq \mathbb{E}$ is a nonempty convex set. We assume that this problem is well-posed in the sense that it admits a solution.

For measuring distances in the space $\mathbb{E}$, we will use the Euclidean seminorm $\|\cdot\|_B$, where $B: \mathbb{E} \rightarrow \mathbb{E}^*$ is a fixed self-adjoint positive semidefinite linear operator.

Our main assumptions on problem (2.1) are as follows. First, we assume that the objective function is consistent with the seminorm, in the sense that there exists a point $\hat{x}_0 \in Q$ and a constant $\gamma_0 > 0$ such that

$$f(x) \geq \gamma_0 \|x - \hat{x}_0\|_B^2, \quad \forall x \in Q. \tag{2.2}$$

Second, we assume we have access to a stochastic gradient oracle for the objective function, specified by a random variable $\xi \sim P_\xi$ taking values in a certain set S_ξ and a mapping $g: \mathbb{E} \times S_\xi \rightarrow \mathbb{E}^*$. We assume that the stochastic gradient oracle may be biased but, on average, the corresponding bias is uniformly bounded in relative scale: there exists $\delta \in [0, 1)$ such that

$$f(y) \geq (1 - \delta)f(x) + \langle \mathcal{E}_\xi[g(x, \xi)], y - x \rangle, \quad \forall x, y \in \mathbb{E}. \quad (2.3)$$

We also assume that the magnitude of stochastic subgradients is relatively bounded w.r.t. the function f : for some $L > 0$,

$$\mathcal{E}_\xi[(\|g(x, \xi)\|_B^*)^2] \leq 2Lf(x), \quad \forall x \in \mathbb{E}. \quad (2.4)$$

The point x_0 and the constants γ_0 , δ and L are supposed to be known. Finally, we need the following technical assumption to guarantee that problem (2.1), as well as certain auxiliary subproblems arising in the method, are well-posed.

Assumption 2.1. *The set $Q + \ker B$ is closed.*

Assumption 2.1 is satisfied, in particular, when Q is closed and B is nondegenerate, or when Q is an affine subspace (and B is arbitrary).

The main auxiliary operation in our method will be the following gradient step:

$$T_Q(\bar{x}, g) := \operatorname{argmin}_{x \in Q} \left\{ \langle g, x \rangle + \frac{1}{2} \|x - \bar{x}\|_B^2 \right\}, \quad \bar{x} \in \mathbb{E}, \quad g \in (\ker B)^\perp. \quad (2.5)$$

Note that problem (2.5) may have multiple solutions (when B is degenerate); in this case, we allow $T_Q(\bar{x}, g)$ to be chosen arbitrarily among them. Nevertheless, a solution to (2.5) always exists.

Lemma 2.2. *Under Assumption 2.1, the point $T := T_Q(\bar{x}, g)$ is well-defined for any $\bar{x} \in \mathbb{E}$ and any $g \in (\ker B)^\perp$, in the sense that problem (2.5) has a solution. This point is characterized by the following equivalent optimality conditions:*

$$\langle g + B(T - \bar{x}), x - T \rangle \geq 0, \quad \forall x \in Q, \quad (2.6)$$

$$\langle g, x - T \rangle + \frac{1}{2} \|x - \bar{x}\|_B^2 \geq \frac{1}{2} \|T - \bar{x}\|_B^2 + \frac{1}{2} \|x - T\|_B^2, \quad \forall x \in Q. \quad (2.7)$$

Proof. Let $\phi: \mathbb{E} \rightarrow \mathbb{R}$ be the function $\phi(x) := \langle g, x \rangle + \frac{1}{2} \|x - \bar{x}\|_B^2$. Clearly, ϕ is closed. Also, ϕ is constant along $\ker B$ (i.e., $\phi(x + h) = \phi(x)$ for all $x \in \mathbb{E}$ and all $h \in \ker B$) since so is $\|\cdot\|_B$ (see Section 1.2) and since $g \in (\ker B)^\perp$. Further, it is not difficult to see that the restriction of ϕ onto $(\ker B)^c$ (a complementary subspace to $\ker B$) has bounded sublevel sets as $\|\cdot\|_B$ is a norm on $(\ker B)^c$ (rather than a seminorm, see Section 1.2). In particular, ϕ restricted to any subset of $(\ker B)^c$ also has bounded sublevel sets. Applying now Lemma A.4 (taking into account Assumption 2.1), we conclude that ϕ has a minimizer on Q , and thus the point T is well-defined.

Inequality (2.6) is the standard first-order optimality condition: a point $T \in Q$ is a minimizer of a differentiable convex function ϕ on a convex set Q iff $\langle \nabla \phi(T), x - T \rangle \geq 0$ for all $x \in Q$. Inequality (2.7) is equivalent to (2.6) in view of the identity

$$\langle B(T - \bar{x}), T - x \rangle = \frac{1}{2} \|T - \bar{x}\|_B^2 + \frac{1}{2} \|x - T\|_B^2 - \frac{1}{2} \|x - \bar{x}\|_B^2$$

which can be easily verified directly. $\square$

Let us present our method for finding an approximate solution to problem (2.1) in relative scale.

Algorithm 2.1: Gradient Method with Relatively Inexact Stochastic Oracle
Input: Stochastic oracle g , initial point $x_0 \in Q$, constants $L, \delta > 0$.
1. Set $v_0 := x_0$, $C_0 := 0$ ($\in \mathbb{R}$). 2. Iterate for $k \geq 0$: a) Compute stochastic gradient $g_k := g(v_k, \xi_k)$, where $\xi_k \sim P_\xi$. b) Choose step size $a_k \in (0, (1 - \delta)/L)$ in a deterministic way. c) Compute coefficients $c_k := a_k(1 - \delta - La_k)$, $C_{k+1} := C_k + c_k$ and the new output point $x_{k+1} := (C_k x_k + c_k v_k)/C_{k+1}$. d) Update prox center $v_{k+1} := T_Q(v_k, a_k g_k)$.

Algorithm 2.1 constructs a sequence of random points $(x_k)_{k=1}^\infty$ each of which depends on the realization of i.i.d. random variables $\xi_1, \dots, \xi_k$. Recall from (2.5) that, in order for Step 2d in this method to be well-defined, the stochastic subgradient g_k should belong to the subspace $(\ker B)^\perp$ at each iteration $k \geq 0$. Let us show that this is indeed the case, and follows from assumption (2.4).

Lemma 2.3. *In Algorithm 2.1, we have $g_k \in (\ker B)^\perp$ (a.s.) for all $k \geq 0$. Thus, at each iteration $k \geq 0$, the computation of v_{k+1} is well-defined (a.s.).*

Proof. Let $k \geq 0$ be arbitrary. According to the definition of g_k at Step 2a and (2.4),

$$\mathcal{E}_{\xi_k}(\|g_k\|_B^*)^2 = \mathcal{E}_{\xi_k}(\|g(v_k, \xi_k)\|_B^*)^2 \leq 2Lf(v_k) < +\infty.$$

This means that $\|g_k\|_B^* < +\infty$ (a.s.). Hence, in view of (1.4), $g_k \in (\ker B)^\perp$ (a.s.). Thus, $a_k g_k \in (\ker B)^\perp$ (a.s.), and hence, by Lemma 2.2, v_{k+1} is well-defined (a.s.). $\square$

Observe that, by definition, for each $k \geq 0$, the point x_{k+1} is a convex combination of x_k and v_k . Since the set Q is convex, and $x_0 = v_0 \in Q$ and $v_k \in Q$ (by its definition at Step 2d) for all $k \geq 0$, we therefore have $x_{k+1} \in Q$ for all $k \geq 0$. Thus, the points $(x_k)_{k=1}^\infty$ constructed by Algorithm 2.1 are all feasible.

Let us now establish a general convergence guarantee for Algorithm 2.1, which is valid for any choice of the coefficients $(a_k)_{k=0}^\infty$.

Lemma 2.4. *In Algorithm 2.1, at any iteration $k \geq 1$, we have, for all $x \in Q$,*

$$\left[\sum_{i=0}^{k-1} a_i(1 - \delta - La_i) \right] \mathcal{E}[f(x_k)] \leq \frac{1}{2} \|x - x_0\|_B^2 + \left[\sum_{i=0}^{k-1} a_i \right] f(x). \quad (2.8)$$

Furthermore, if $x_0 = \hat{x}_0$ (where $\hat{x}_0 \in Q$ satisfies (2.2)), then, at any iteration² $k \geq 1$,

$$(1 - \delta_k) \mathcal{E}[f(x_k)] \leq f^*, \quad \text{where} \quad \delta_k := \frac{1 + 2\gamma_0 \sum_{i=0}^{k-1} a_i(\delta + La_i)}{1 + 2\gamma_0 \sum_{i=0}^{k-1} a_i} \quad (< 1). \quad (2.9)$$

²The fact that $\delta_k < 1$ follows from our assumption at Step 2b.

Proof. i. Let $x \in Q$ and $k \geq 0$ be arbitrary. By the definition of v_{k+1} at Step 2d and Lemma 2.2, we have

$$a_k \langle g_k, x - v_{k+1} \rangle + \frac{1}{2} \|x - v_k\|_B^2 \geq \frac{1}{2} \|v_{k+1} - v_k\|_B^2 + \frac{1}{2} \|x - v_{k+1}\|_B^2.$$

Rearranging and using (1.5), we obtain

$$\begin{aligned} \frac{1}{2} \|x - v_{k+1}\|_B^2 - \frac{1}{2} \|x - v_k\|_B^2 &\leq a_k \langle g_k, x - v_{k+1} \rangle - \frac{1}{2} \|v_{k+1} - v_k\|_B^2 \\ &= a_k \langle g_k, x - v_k \rangle + a_k \langle g_k, v_{k+1} - v_k \rangle - \frac{1}{2} \|v_{k+1} - v_k\|_B^2 \\ &\leq a_k \langle g_k, x - v_k \rangle + \frac{1}{2} a_k^2 (\|g_k\|_B^*)^2. \end{aligned}$$

Recall that $g_k = g(v_k, \xi_k)$ (see Step 2a). Further, by the construction of Algorithm 2.1, a_k is deterministic, while v_k and x_{k+1} are independent of ξ_{k+1} . Therefore, passing to expectations w.r.t. ξ_{k+1} in the above display and using (2.3) and (2.4), we get

$$\begin{aligned} \frac{1}{2} \mathcal{E}_{\xi_k} [\|x - v_{k+1}\|_B^2] - \frac{1}{2} \|x - v_k\|_B^2 &\leq a_k [f(x) - (1 - \delta)f(v_k)] + L a_k^2 f(v_k) \\ &= a_k f(x) - c_k f(v_k). \end{aligned}$$

(The final identity follows from the definition of c_k at Step 2c.) Passing to full expectations and rearranging, we get

$$\mathcal{E}[c_k f(v_k)] + \frac{1}{2} \mathcal{E}[\|x - v_{k+1}\|_B^2] \leq a_k f(x) + \frac{1}{2} \mathcal{E}[\|x - v_k\|_B^2].$$

Note that this inequality is valid for any $k \geq 0$.

Summing up the above inequalities for all indices $0 \leq k' \leq k - 1$, where $k \geq 1$ is arbitrary, dropping the term $\mathcal{E}[\|x - v_k\|_B^2] \geq 0$ and recalling that $v_0 = x_0$, we obtain

$$\mathcal{E} \left[\sum_{i=0}^{k-1} c_i f(v_i) \right] \leq \left[\sum_{i=0}^{k-1} a_i \right] f(x) + \frac{1}{2} \|x - x_0\|_B^2.$$

On the other hand, from the definitions at Step 2c, it follows that $C_k x_k = \sum_{i=0}^{k-1} c_i v_i$ with

$$C_k = \sum_{i=0}^{k-1} c_i = \sum_{i=0}^{k-1} a_i (1 - \delta - L a_i) \quad (2.10)$$

being a deterministic coefficient (since each a_i is assumed to be so). Hence, by the convexity of f , the left-hand side in the above display is $\geq C_k \mathcal{E}[f(x_k)]$. Substituting further (2.10), we obtain (2.8).

ii. Let us prove (2.9). Let $k \geq 1$ and $x \in Q$ be arbitrary. Putting together (2.8) and (2.2), we get

$$C_k \mathcal{E}[f(x_k)] \leq \left(\frac{1}{2\gamma_0} + \sum_{i=0}^{k-1} a_i \right) f(x).$$

Hence, according to (2.10),

$$\frac{C_k}{\frac{1}{2\gamma_0} + \sum_{i=0}^{k-1} a_i} = \frac{\sum_{i=0}^{k-1} a_i(1 - \delta - La_i)}{\frac{1}{2\gamma_0} + \sum_{i=0}^{k-1} a_i} = 1 - \frac{1 + 2\gamma_0 \sum_{i=0}^{k-1} a_i(\delta + La_i)}{1 + 2\gamma_0 \sum_{i=0}^{k-1} a_i} = 1 - \delta_k$$

Combining the above two displays, we get $(1 - \delta_k)\mathcal{E}[f(x_k)] \leq f(x)$. This proves (2.9) in view of the definition of f^* in (2.1) and the fact that $x \in Q$ was arbitrary. $\square$

Thus, after $k \geq 1$ iterations, Algorithm 2.1 generates a point $x_k \in Q$ which is, on average, a δ_k -approximate solution to (2.1) in relative scale. Let us show that, by appropriately choosing step sizes a_k in the method, we can make δ_k sufficiently small (for a sufficiently large k).

First, observe that, for any $k \geq 0$, we have

$$\delta_k = \delta + \frac{1 - \delta + 2\gamma_0 L \sum_{i=0}^{k-1} a_i^2}{1 + 2\gamma_0 \sum_{i=0}^{k-1} a_i} \geq \delta. \quad (2.11)$$

Therefore, we cannot hope for Algorithm 2.1 to produce an approximation solution whose relative accuracy will be better than that of the oracle itself. This is quite natural. At the same time, we can easily ensure that $\delta_k \rightarrow \delta$ as $k \rightarrow \infty$. For this, it suffices to choose step sizes a_k in such a way that

$$\sum_{k=0}^{\infty} a_k = \infty, \quad \sum_{k=0}^{\infty} a_k^2 < \infty,$$

which is a standard recipe for subgradient methods (see, e.g., Section 3.2.3 in [5]).

Let us now derive an optimal choice of step sizes for Algorithm 2.1. This is easier when we fix the total number of steps, say, $N \geq 1$. From (2.11), it is not difficult to see that δ_N is a symmetric convex function of $(a_i)_{i=0}^{N-1}$. Hence, its minimum is attained at $a_i = a_N^*$, $0 \leq i \leq N-1$, where a_N^* minimizes the ratio

$$\delta_N(a) := \frac{1 + 2\gamma_0 Na(\delta + La)}{1 + 2\gamma_0 Na} \quad (2.12)$$

over all $a \in (0, \frac{1-\delta}{L})$. Differentiating $\delta_N(a)$ in a and setting the derivative to zero, we come to the following equation for a_N^* :

$$(\delta + 2La_N^*)(1 + 2\gamma_0 Na_N^*) = 1 + 2\gamma_0 Na_N^*(\delta + La_N^*). \quad (2.13)$$

This is the quadratic equation $2\gamma_0 NL(a_N^*)^2 + 2La_N^* = 1 - \delta$ with a unique positive solution

$$a_N^* = \frac{\sqrt{2\gamma_0 NL(1 - \delta) + L^2} - L}{2\gamma_0 NL} = \frac{1 - \delta}{\sqrt{2\gamma_0 NL(1 - \delta) + L^2} + L} \quad \left(< \frac{1 - \delta}{L} \right). \quad (2.14)$$

Substituting this value into (2.12) and taking into account (2.13), we obtain

$$\delta_N(a_N^*) - \delta = 2La_N^* = \frac{2\sqrt{L}(1 - \delta)}{\sqrt{2\gamma_0 N(1 - \delta) + L} + \sqrt{L}} \leq \sqrt{\frac{2L}{\gamma_0 N}}.$$

Thus, for the optimal choice of the step size (2.14), we have the $O(1/\sqrt{N})$ convergence rate to the level δ . In particular, for generating a point $x_N \in Q$ such that

$$(1 - 2\delta)\mathcal{E}[f(x_N)] \leq f^*,$$

it suffices to make

$$N \geq N(\delta) := \frac{2L}{\gamma_0\delta^2} \quad (2.15)$$

iterations of Algorithm 2.1 with step sizes (2.14).

Instead of the optimal step sizes (2.14), we can use another (simpler) choice that leads to the same complexity guarantee (2.15) but requires only the knowledge of δ and L .

Theorem 2.5. *Consider Algorithm 2.1 with constant step sizes*

$$a_k := \frac{\delta}{2L} \quad \left(< \frac{1}{L} \right), \quad k \geq 0, \quad (2.16)$$

and the initial point $x_0 = \hat{x}_0$ (where $\hat{x}_0 \in Q$ satisfies (2.2)). Then, for any integer

$$N \geq N(\delta) := \frac{2L}{\gamma_0\delta^2}, \quad (2.17)$$

we have

$$(1 - 2\delta)\mathcal{E}[f(x_N)] \leq f^*.$$

Proof. Substituting (2.16) into (2.11), we obtain, for any $N \geq N(\delta)$,

$$\begin{aligned} \delta_N - \delta &= \frac{1 - \delta + 2\gamma_0 L N \frac{\delta^2}{4L^2}}{1 + 2\gamma_0 N \frac{\delta}{2L}} = \frac{2L(1 - \delta) + \gamma_0 N \delta^2}{2L + 2\gamma_0 N \delta} \\ &\leq \frac{L}{\gamma_0 N \delta} + \frac{\delta}{2} \leq \frac{L}{\gamma_0 N(\delta) \delta} + \frac{\delta}{2} = \delta, \end{aligned}$$

where the final identity follows from (2.17). It remains to apply Lemma 2.4. $\square$

2.2 Dual Averaging Method

The Gradient Method from Section 2.1 has a couple of significant drawbacks. First, it uses the same step size at every iteration, which is proportional to the desired accuracy level (formula (2.16)). This requires the user to know in advance the final accuracy they want to obtain, and means that the method is essentially a *short-step* one—its step size is always small no matter what. Instead, it would be more natural to start with a large step size at the initial iterations, and then gradually decrease it.

Second, the algorithm works with an oracle whose accuracy is assumed to be fixed. However, in many cases, this quantity is actually a parameter of the oracle, and one can query the oracle at the same point with various accuracies (see Section 3 for examples). Since the complexity of an oracle is usually monotone in the required accuracy, it makes sense to start with a large inaccuracy level and then gradually decrease it in iterations, similarly to the above discussion on the step sizes.

In this section, we present an algorithm that addresses the above drawbacks and still enjoys the same worst-case complexity guarantee as the method from Section 2.1. Our algorithm is based on the Dual Averaging method from [4].

Our setup is almost the same as in Section 2.1. Specifically, we are interested in solving the problem (2.1) under the assumptions (2.2), (2.3) and (2.4). The only difference is that now we assume that δ is also an input of the oracle g , so we should write $g(\delta, x, \xi)$ instead of $g(x, \xi)$. (For simplicity, we assume that the constant L in (2.4) is independent of δ .)

We consider the following method:

Algorithm 2.2: Dual Averaging with Relatively Inexact Stochastic Oracle
Input: Oracle g , initial point $x_0 \in Q$, constant L , deterministic positive sequences $(a_k)_{k=1}^\infty$, $(\beta_k)_{k=0}^\infty$ and $(\delta_k)_{k=1}^\infty$.
Preconditions: $(\beta_k)_{k=0}^\infty$ is non-decreasing; $\frac{La_k}{\beta_k} < 1 - \delta_k$ for all $k \geq 1$.
1. $v_0 := x_0, \bar{g}_0 := 0$ ($\in \mathbb{E}^*$), $A_0 := C_0 := 0$ ($\in \mathbb{R}$). 2. Iterate for $k \geq 0$: a) $w_{k+1} := (\beta_k v_k + (\beta_{k+1} - \beta_k)x_0)/\beta_{k+1}$. b) $g_{k+1} := g(\delta_{k+1}, w_{k+1}, \xi_{k+1})$ for an i.i.d. sampled ξ_{k+1} . c) $A_{k+1} := A_k + a_{k+1}, \bar{g}_{k+1} := (A_k \bar{g}_k + a_{k+1} g_{k+1})/A_{k+1}$. d) $v_{k+1} := T_Q(x_0, \frac{A_{k+1}}{\beta_{k+1}} \bar{g}_{k+1})$. e) $c_{k+1} := a_{k+1}(1 - \delta_{k+1} - \frac{La_{k+1}}{\beta_{k+1}})$, $C_{k+1} := C_k + c_{k+1}$, $x_{k+1} := (C_k x_k + c_{k+1} w_{k+1})/C_{k+1}$.

Theorem 2.6. Consider Algorithm 2.2 with $x_0 = \hat{x}_0$ (see (2.2)). Then, for any $k \geq 1$,

$$(1 - \Delta_k) \mathcal{E}f(x_k) \leq f^*, \quad \text{where} \quad \Delta_k := \frac{\beta_k + 2\gamma_0 \sum_{i=1}^k a_i (\delta_i + \frac{La_i}{\beta_i})}{\beta_k + 2\gamma_0 A_k} \quad (< 1). \quad (2.18)$$

Proof. Let us define, for any $k \geq 0$, the following “estimating function” $\psi_k: \mathbb{E} \rightarrow \mathbb{R}$:

$$\psi_k(x) := \frac{\beta_k}{2} \|x - x_0\|_B^2 + \sum_{i=1}^k a_i \ell_i(x), \quad (2.19)$$

where, for any $k \geq 1$,

$$\ell_k(x) := (1 - \delta_k) f(w_k) + \langle g_k, x - w_k \rangle.$$

Note that, according to (2.3), in expectation, the function ℓ_k is a lower bound on f over Q :

$$\mathcal{E}_{\xi_k} \ell_k(x) = (1 - \delta_k) f(w_k) + \langle \mathcal{E}_{\xi_k} g_k, x - w_k \rangle \leq f(x), \quad \forall x \in Q.$$

Therefore, for any $k \geq 1$ and any $x \in Q$, we have

$$\mathcal{E} \psi_k(x) \leq \frac{\beta_k}{2} \|x - x_0\|_B^2 + A_k f(x). \quad (2.20)$$

Let us show that, for any $k \geq 0$, we have

$$v_k = \operatorname{argmin}_{x \in Q} \psi_k(x).$$

This is obvious for $k = 0$ since, by our definition, $v_0 = x_0 \in Q$. Let $k \geq 1$. From the definitions at Steps 1 and 2c, it follows that

$$\bar{g}_k = \frac{1}{A_k} \sum_{i=1}^k a_i g_i, \quad A_k = \sum_{i=1}^k a_i,$$

Hence, according to its definition at Step 2d, for any $k \geq 1$, we have

$$v_k = T_Q\left(x_0, \frac{A_k}{\beta_k} \bar{g}_k\right) = T_Q\left(x_0, \frac{1}{\beta_k} \sum_{i=1}^k a_i g_i\right) = \operatorname{argmin}_{x \in Q} \psi_k(x),$$

where the final identity is due to (2.5).

Since v_k is the minimizer of ψ_k over Q and ψ_k is a 1-strongly convex function, we have, for any $k \geq 0$ and any $x \in Q$,

$$\psi_k(x) \geq \psi_k^* + \frac{\beta_k}{2} \|x - v_k\|_B^2,$$

where $\psi_k^* := \psi_k(v_k)$ be the minimal value of ψ_k on Q .

Let $k \geq 0$ be an arbitrary index. According to (2.19) and the above display, for any $x \in Q$, we have

$$\begin{aligned} \psi_{k+1}(x) &= \psi_k(x) + \frac{\beta_{k+1} - \beta_k}{2} \|x - x_0\|_B^2 + a_{k+1} \ell_{k+1}(x) \\ &\geq \psi_k^* + \frac{\beta_k}{2} \|x - v_k\|_B^2 + \frac{\beta_{k+1} - \beta_k}{2} \|x - x_0\|_B^2 + a_{k+1} \ell_{k+1}(x) \\ &\geq \psi_k^* + \frac{\beta_{k+1}}{2} \|x - w_{k+1}\|_B^2 + a_{k+1} \ell_{k+1}(x), \end{aligned}$$

where the final inequality follows from the convexity of the squared (semi)norm and the definition of w_{k+1} at Step 2a (note that, according to our requirements, $\beta_k \leq \beta_{k+1}$). Substituting now $x = v_{k+1}$ together with the definition of $\ell_{k+1}(x)$, we get

$$\begin{aligned} \psi_{k+1}^* - \psi_k^* &\geq \frac{\beta_{k+1}}{2} \|v_{k+1} - w_{k+1}\|_B^2 + a_{k+1} [(1 - \delta_{k+1})f(w_{k+1}) + \langle g_{k+1}, v_{k+1} - w_{k+1} \rangle] \\ &\geq a_{k+1} \left[(1 - \delta_{k+1})f(w_{k+1}) - \frac{a_{k+1}}{2\beta_{k+1}} (\|g_{k+1}\|_B^*)^2 \right]. \end{aligned}$$

Taking the expectation w.r.t. ξ_{k+1} , using the fact that $\mathcal{E}_{\xi_{k+1}}[(\|g_{k+1}\|_B^*)^2] \leq 2Lf(w_{k+1})$ (see (2.4)), and the definition of c_{k+1} at Step 2e, we obtain

$$\mathcal{E}_{\xi_{k+1}} \psi_{k+1}^* - \psi_k^* \geq a_{k+1} \left(1 - \delta_{k+1} - \frac{La_{k+1}}{\beta_{k+1}} \right) f(w_{k+1}) = c_{k+1} f(w_{k+1}).$$

Passing to full expectations in the above inequalities, summing up and using the fact that $\psi_0^* = 0$, we get, for any $k \geq 1$,

$$\mathcal{E} \psi_k^* \geq \sum_{i=1}^k c_i \mathcal{E} f(w_i) \geq C_k \mathcal{E} f(x_k), \quad (2.21)$$

where the final inequality follows from the convexity of f and the fact that

$$x_k = \frac{1}{C_k} \sum_{i=1}^k c_i w_i, \quad C_k = \sum_{i=1}^k c_i$$

(see the definitions at Steps 1 and 2e).

Combining (2.21) with (2.20) and (2.2), we finally obtain, for any $x \in Q$,

$$C_k \mathcal{E}f(x_k) \leq \mathcal{E}\psi_k^* \leq \mathcal{E}\psi_k(x) \leq \frac{\beta_k}{2} \|x - x_0\|_B^2 + A_k f(x) \leq \left(\frac{\beta_k}{2\gamma_0} + A_k \right) f(x).$$

This means that

$$(1 - \Delta_k) \mathcal{E}f(x_k) \leq f^*,$$

where

$$\begin{aligned} \Delta_k &:= 1 - \frac{C_k}{\frac{\beta_k}{2\gamma_0} + A_k} = 1 - \frac{\sum_{i=1}^k a_i (1 - \delta_i - \frac{La_i}{\beta_i})}{\frac{\beta_k}{2\gamma_0} + A_k} \\ &= \frac{\frac{\beta_k}{2\gamma_0} + \sum_{i=1}^k a_i (\delta_i + \frac{La_i}{\beta_i})}{\frac{\beta_k}{2\gamma_0} + A_k} = \frac{\beta_k + 2\gamma_0 \sum_{i=1}^k a_i (\delta_i + \frac{La_i}{\beta_i})}{\beta_k + 2\gamma_0 A_k}. \end{aligned}$$

Note that $\Delta_k < 1$ in view of our assumption that $\frac{La_k}{\beta_k} \leq 1 - \delta_k$ for all $i \geq 1$. $\square$

Comparing the convergence rate estimate (2.18) with the corresponding estimate (2.9) for Algorithm 2.1, we see that they are very similar. However, the former is more flexible. First, it contains an additional control sequence β_k which can be chosen by the user. Second, the oracle inaccuracy δ is allowed to vary at each iteration. Note that, by choosing $\beta_k \equiv 1$ and $\delta_k \equiv \delta$ in Algorithm 2.2, we obtain exactly the same estimate as for Algorithm 2.1.

The particular form of the convergence rate estimate (2.18) suggests the following choice of oracle inaccuracies:

$$\delta_k := \frac{La_k}{\beta_k}, \quad k \geq 1. \quad (2.22)$$

This choice is natural since then both terms δ_i and $\frac{La_i}{\beta_i}$ in (2.18) are well-balanced, while the corresponding expression for Δ_k is exactly the same, up to an absolute constant, as if we had $\delta_i \equiv 0$:

$$\Delta_k = \frac{\beta_k + 4\gamma_0 L \sum_{i=1}^k \frac{a_i^2}{\beta_i}}{\beta_k + 2\gamma_0 A_k}.$$

Note that the above expression is very similar to that which arises in the usual convergence analysis of the standard Dual Averaging method (see [4]). In particular, dropping β_k from the denominator in the above display, we obtain

$$\Delta_k \leq \frac{\beta_k}{2\gamma_0 A_k} + \frac{2L}{A_k} \sum_{i=1}^k \frac{a_i^2}{\beta_i}.$$

Thus, we can try to use the same strategies for choosing the coefficients a_k and β_k as in the standard Dual Averaging method from [4]. The simplest one is to choose

$$a_k = 1, \quad k \geq 1, \quad (2.23)$$

and $\beta_k \sim \sqrt{\gamma_0 L k}$. However, we still need to respect the constraint that $\frac{La_k}{\beta_k} < 1 - \delta_k$ for all $k \geq 1$ (see the preconditions in Algorithm 2.2). For our choice of a_k and δ_k , this means $\beta_k > 2L$ for all $k \geq 1$. It is therefore reasonable to choose

$$\beta_k = \sqrt{8\gamma_0 L k} + 2L, \quad k \geq 0. \quad (2.24)$$

(The particular value of the absolute constant under the square root gives a “reasonably small” absolute constant in the final iteration complexity bound below.)

Theorem 2.7. *Consider Algorithm 2.2 with coefficients given by (2.22)–(2.24). Then, for any $\delta \in (0, 1)$, we have*

$$(1 - \delta)\mathcal{E}f(x_k) \leq f^*$$

whenever $k \geq N(\delta)$, where

$$N(\delta) := \frac{10L}{\gamma_0 \delta^2}.$$

Proof. According to (2.24), $(\beta_k)_{k=0}^\infty$ is an increasing sequence with $\beta_k > 2L$ for all $k \geq 1$. Consequently, for any $k \geq 1$, we have $\frac{La_k}{\beta_k} + \delta_k = \frac{2L}{\beta_k} < 1$, so the preconditions of Algorithm 2.2 are satisfied.

Let Δ_k be defined as in Theorem 2.6. We need to show that $\Delta_k \leq \delta$ for any $k \geq N(\delta)$.

Let $k \geq 1$ be arbitrary. According to (2.22) and (2.23), we have $A_k = k$, and

$$\Delta_k = \frac{\beta_k + 2\gamma_0 \sum_{i=1}^k a_i(\delta_i + \frac{La_i}{\beta_i})}{\beta_k + 2\gamma_0 A_k} = \frac{\beta_k + 4\gamma_0 L \sum_{i=1}^k \frac{1}{\beta_i}}{\beta_k + 2\gamma_0 k} \leq \frac{\beta_k + 4\gamma_0 L \sum_{i=1}^k \frac{1}{\beta_i}}{2\gamma_0 k}.$$

Note that

$$\sum_{i=1}^k \frac{1}{\beta_i} = \sum_{i=1}^k \frac{1}{\sqrt{8\gamma_0 L i} + 2L} \leq \frac{1}{\sqrt{8\gamma_0 L}} \sum_{i=1}^k \frac{1}{\sqrt{i}} \leq \sqrt{\frac{k}{2\gamma_0 L}},$$

where the final inequality follows from the fact that $\sum_{i=1}^k \frac{1}{\sqrt{i}} \leq \int_0^k \frac{dt}{\sqrt{t}} = 2\sqrt{k}$. Thus,

$$\Delta_k \leq \frac{\sqrt{8\gamma_0 L k} + 2L + 4\gamma_0 L \sqrt{\frac{k}{2\gamma_0 L}}}{2\gamma_0 k} = \frac{2\sqrt{8\gamma_0 L k} + 2L}{2\gamma_0 k} = \sqrt{\frac{8L}{\gamma_0 k}} + \frac{L}{\gamma_0 k}.$$

Let $\delta \in (0, 1)$ be arbitrary. Denote $\tau_k := \sqrt{\frac{L}{\gamma_0 k}}$. According to the above estimate, for any $k \geq 1$, we have $\Delta_k \leq 2\sqrt{2}\tau_k + \tau_k^2$. To ensure that $\Delta_k \leq \delta$, it therefore suffices to ensure that $\tau_k \leq \tau$, where τ is the positive root of the following equation:

$$2\sqrt{2}\tau + \tau^2 = \delta.$$

Solving this quadratic equation, we obtain

$$\tau = \sqrt{2 + \delta} - \sqrt{2} = \frac{\delta}{\sqrt{2 + \delta} + \sqrt{2}} \geq \frac{\delta}{\sqrt{3} + \sqrt{2}}.$$

Thus, it suffices to ensure that $\tau_k \leq \frac{\delta}{\sqrt{3}+\sqrt{2}}$, or, equivalently, that

$$k \geq (\sqrt{3} + \sqrt{2})^2 \frac{L}{\gamma_0 \delta^2} = (5 + 2\sqrt{6}) \frac{L}{\gamma_0 \delta^2}.$$

It remains to note that $5 + 2\sqrt{6} \leq 10$. $\square$

Thus, Algorithm 2.2 with coefficients given by (2.22)–(2.24) has the same worst-case iteration complexity (up to an absolute constant) as Algorithm 2.1 with the fixed oracle inaccuracy δ . Note, however, that, in our new method, the oracle inaccuracy δ_k decreases at the following rate:

$$\delta_k = \frac{L}{\sqrt{8\gamma_0 Lk} + 2L} = \frac{1}{\sqrt{\frac{8\gamma_0 k}{L}} + 2} \sim \sqrt{\frac{L}{\gamma_0 k}}.$$

In particular, for all $k \leq N(\delta) \sim \frac{L}{\gamma_0 \delta^2}$, we have $\delta_k \gtrsim \delta$ which means that the new algorithm never needs the oracle inaccuracy less than δ (up to an absolute constant). For any reasonable oracle (whose complexity is polynomial in δ), it means that the worst-case total complexity of the Dual Averaging method is the same (in order) as that of the previously considered Gradient Method with fixed step sizes. If, however, the actual iteration complexity $k(\delta)$ of the new algorithm turns out to be significantly smaller than the worst-case $N(\delta)$ on some particular problem, then the total oracle complexity of the new algorithm may be significantly smaller as well.

2.3 Composition with Affine Mapping

Let us show that our main assumptions from Section 2.1 are preserved when composing a function with an affine mapping.

Consider the problem

$$\min_{x \in Q} [f(x) := F(Ax + b)], \quad (2.25)$$

where $Q \subseteq \mathbb{E}$ is a nonempty convex set, $A: \mathbb{E} \rightarrow \mathbb{E}_1$ is a linear operator, $b \in \mathbb{E}_1$, and $F: \mathbb{E}_1 \rightarrow \mathbb{R}$ is a convex function satisfying assumptions from Section 2.1 on the set

$$Q_1 := A(Q) + b \quad (\subseteq \mathbb{E}_1). \quad (2.26)$$

Specifically, we assume the space $\mathbb{E}_1$ is equipped with a certain Euclidean seminorm $\|\cdot\|_{B_1}$, where $B_1: \mathbb{E}_1 \rightarrow \mathbb{E}_1^*$ is a self-adjoint positive semidefinite linear operator, and that the function F is consistent with this seminorm:

$$F(y) \geq \gamma_0 \|y - \hat{y}_0\|_{B_1}^2, \quad \forall y \in Q_1, \quad (2.27)$$

where $\hat{y}_0 \in \mathbb{E}_1$ and $\gamma_0 > 0$. We assume the following regularity condition is satisfied.

Assumption 2.8. *The set $Q_1 + \ker B_1$ is closed.*

Assumption 2.8 is satisfied, in particular, when Q is an affine subspace (e.g., when problem (2.25) is unconstrained, i.e., $Q = \mathbb{E}$).

We also assume that the function F is represented by a relatively inexact stochastic gradient oracle (G, ξ) with relatively bounded magnitude: for some $\delta \in [0, 1)$ and $L > 0$,

$$F(y) \geq (1 - \delta)F(y_1) + \langle \mathcal{E}_\xi[G(y_1, \xi)], y - y_1 \rangle, \quad \forall y, y_1 \in \mathbb{E}_1, \quad (2.28)$$

$$\mathcal{E}_\xi[(\|G(y, \xi)\|_{B_1}^*)^2] \leq 2LF(y), \quad \forall y \in \mathbb{E}_1. \quad (2.29)$$

A natural stochastic oracle for the function f is, of course, (g, ξ) defined by

$$g(x, \xi) := A^*G(Ax + b, \xi), \quad x \in \mathbb{E}. \quad (2.30)$$

It is not difficult to see that this oracle is δ -relatively approximate for f : for any $x, y \in \mathbb{E}$, we have, according to our assumption (2.28),

$$\begin{aligned} f(y) &= F(Ay + b) \geq (1 - \delta)F(Ax + b) + \langle \mathcal{E}_\xi[G(Ax + b, \xi)], A(y - x) \rangle \\ &= (1 - \delta)f(x) + \langle \mathcal{E}_\xi[g(x, \xi)], y - x \rangle. \end{aligned}$$

Now let us introduce a Euclidean seminorm in the space $\mathbb{E}$. A good choice is

$$\|x\|_B := \|Ax\|_{B_1}, \quad x \in \mathbb{E}, \quad (2.31)$$

which is the seminorm induced by the operator

$$B := A^*B_1A. \quad (2.32)$$

This choice is good for several reasons. First, it correctly “translates” our closedness assumption 2.8 from the space $\mathbb{E}_1$ into $\mathbb{E}$.

Lemma 2.9. *Under Assumption 2.8, the set $Q + \ker B$ is closed.*

Proof. Note from (2.32) that $\ker B = \ker(B_1A)$ (B_1 is positive semidefinite). Combining this with (2.26) and the fact that closedness is a translation-invariant property, we see that we need to prove the following implication:

$$A(Q) + \ker B_1 \text{ is closed} \implies Q + \ker(B_1A) \text{ is closed.}$$

But this follows from Lemma A.1. □

Second, our choice of the seminorm preserves the consistency constants γ_0 and L . Indeed, let us define $\hat{x}_0$ in the following way:

$$\hat{x}_0 := \operatorname{argmin}_{x \in Q} \|Ax + b - \hat{y}_0\|_{B_1}^2 = T_Q(0, A^*B_1(b - \hat{y}_0)), \quad (2.33)$$

where T_Q is the gradient step defined in (2.5) (w.r.t. the seminorm $\|\cdot\|_B$ with B given by (2.32)). The identity in (2.33) follows from the fact that

$$\begin{aligned} \|Ax + b - \hat{y}_0\|_{B_1}^2 &= \|Ax\|_{B_1}^2 + \langle B_1Ax, b - \hat{y}_0 \rangle + \|b - \hat{y}_0\|_{B_1}^2 \\ &= \|x\|_B^2 + \langle A^*B_1(b - \hat{y}_0), x \rangle + \|b - \hat{y}_0\|_{B_1}^2. \end{aligned}$$

Observe that $A^*B_1(b - \hat{y}_0) \in (\ker(B_1A))^\perp = (\ker B)^\perp$, hence, according to Lemma 2.2, the point $\hat{x}_0$ is well-defined.

Lemma 2.10. *It holds that*

$$f(x) \geq \gamma_0 \|x - \hat{x}_0\|_B^2, \quad \forall x \in Q. \quad (2.34)$$

$$\mathcal{E}_\xi[(\|g(x, \xi)\|_B^*)^2] \leq 2Lf(x), \quad \forall x \in \mathbb{E}. \quad (2.35)$$

Proof. Let us prove (2.34). Let $x \in Q$ be arbitrary. From (2.25) and (2.27), we get

$$f(x) = F(Ax + b) \geq \gamma_0 \|Ax + b - \hat{y}_0\|_{B_1}^2.$$

It remains to prove that $\|Ax + b - \hat{y}_0\|_{B_1} \geq \|x - \hat{x}_0\|_B$, or, more generally, that

$$\|Ax + b - \hat{y}_0\|_{B_1}^2 \geq \|A\hat{x}_0 + b - \hat{y}_0\|_{B_1}^2 + \|x - \hat{x}_0\|_B^2. \quad (2.36)$$

This follows from (2.33). Indeed, by Lemma 2.2, we have

$$2\langle A^*B_1(b - \hat{y}_0), x - \hat{x}_0 \rangle + \|x\|_B^2 \geq \|\hat{x}_0\|_B^2 + \|x - \hat{x}_0\|_B^2.$$

Rearranging and using (2.31), we can rewrite this as follows:

$$2\langle B_1Ax, b - \hat{y}_0 \rangle + \|Ax\|_{B_1}^2 \geq 2\langle B_1A\hat{x}_0, b - \hat{y}_0 \rangle + \|A\hat{x}_0\|_{B_1}^2 + \|x - \hat{x}_0\|_B^2.$$

Adding $\|b - \hat{y}_0\|_{B_1}^2$ to both sides and completing the squares, we get (2.36).

Let us prove (2.35). Let $x \in \mathbb{E}$ be arbitrary. According to (1.3), (2.30) and (2.31),

$$\begin{aligned} \|g(x, \xi)\|_B^* &= \sup_{h \in \mathbb{E}} \{ \langle g(x, \xi), h \rangle : \|h\|_B \leq 1 \} \\ &= \sup_{h \in \mathbb{E}} \{ \langle G(Ax + b, \xi), Ah \rangle : \|Ah\|_{B_1} \leq 1 \} \leq \|G(Ax + b, \xi)\|_{B_1}^*. \end{aligned}$$

Combining this with (2.29) and (2.25), we obtain

$$\mathcal{E}_\xi[(\|g(x, \xi)\|_B^*)^2] \leq \mathcal{E}_\xi[(\|G(Ax + b, \xi)\|_{B_1}^*)^2] \leq 2LF(Ax + b) = 2Lf(x). \quad \square$$

3 Applications in Semidefinite Optimization

Let us present several examples of relatively inexact stochastic subgradient oracles, which are suitable for various functions of matrix eigen- and singular values.

In this section, our random vectors belong to $\mathbb{R}^n$. Therefore, we will use notation $\|\cdot\|$ for the standard Euclidean norm.

3.1 Approximating Eigenvectors and Singular Vectors

3.1.1 Relatively Inexact Eigen- And Singular Vectors

Definition 3.1 (Relatively inexact maximal eigenvector). Given a matrix $A \in \mathbb{S}_+^n$, a scalar $\delta \in (0, 1)$ and a random vector $\hat{v} \in \mathbb{R}^n$, we say that $\hat{v}$ is a δ -relatively inexact stochastic maximal unit eigenvector of A iff $\|\hat{v}\| = 1$ (a.s.) and

$$\mathcal{E}\langle A\hat{v}, \hat{v} \rangle \geq (1 - \delta)\lambda_{\max}(A).$$

A *relatively inexact stochastic maximal eigenvector oracle* is a procedure MaxEV that takes a matrix $A \in \mathbb{S}_+^n$ and a scalar $\delta \in (0, 1)$, and returns a random vector $\hat{v} = \text{MaxEV}(A, \delta)$ such that $\hat{v}$ is a δ -relatively inexact stochastic maximal unit eigenvector of A .

Sometimes, it is necessary to have some guarantees not only for the Rayleigh quotient $\langle A\hat{v}, \hat{v} \rangle$ but also for its powers. Since the expectation is not, in general, invariant w.r.t. taking powers, it makes sense to introduce the following generalization of Definition 3.1.

Definition 3.2. Let $A \in \mathbb{S}_+^n$ be a matrix, and let $\delta \in (0, 1)$ and $p \in (0, \infty)$ be scalars. A random vector $\hat{v} \in \mathbb{R}^n$ is called a δ -relatively inexact stochastic maximal unit eigenvector of A of degree p iff $\|\hat{v}\| = 1$ (a.s.) and

$$\mathcal{E}[\langle A\hat{v}, \hat{v} \rangle^p] \geq (1 - \delta)[\lambda_{\max}(A)]^p. \quad (3.1)$$

A *relatively inexact stochastic maximal eigenvector oracle of degree $p \in (0, \infty)$* is a procedure MaxEV_p that accepts a matrix $A \in \mathbb{S}_+^n$ and a scalar $\delta \in (0, 1)$, and returns a random vector $\hat{v} = \text{MaxEV}_p(A, \delta)$ such that $\hat{v}$ is a δ -relatively inexact stochastic maximal unit eigenvector of A of degree p .

Note that an oracle for $\text{MaxEV}_p(A, \delta)$ can be easily implemented in terms of that for $\text{MaxEV}(A, \delta)$. In what follows, we refer to this implementation as the default one.

Algorithm 3.1		
Signature:	MaxEV_p	$=$ $\text{MakeMaxEVOFdeg}(p, \text{MaxEV}).$
Input:	p —degree [$p \in (0, \infty)$]; MaxEV —relatively inexact stochastic maximal eigenvector oracle.	
Output:	MaxEV_p —relatively inexact stochastic maximal eigenvector oracle of degree p .	
1. If $p \leq 1$, return MaxEV. 2. Otherwise, return the function defined by		
$(A, \delta) \mapsto \text{MaxEV}(A, 1 - (1 - \delta)^{1/p}).$		

Lemma 3.3. *The output in Algorithm 3.1 is indeed a relatively inexact stochastic eigenvector oracle of degree p .*

Proof. Let $A \in \mathbb{S}_+^n$ and $\delta \in (0, 1)$ be arbitrary, and let $\hat{v} := \text{MaxEV}_p(A, \delta)$ be the output of the MaxEV_p procedure constructed by Algorithm 3.1. Let us show that $\hat{v}$ is a δ -relatively inexact stochastic eigenvector of A of degree p .

Clearly, $\|\hat{v}\| = 1$ (a.s.) since $\hat{v}$ is the output of the MaxEV procedure. Therefore, we only need to justify inequality (3.1). We may also assume that $\lambda_{\max}(A) > 0$ since otherwise the inequality is trivial in view of positive semidefiniteness of A .

If $p \leq 1$, then, by construction, $\hat{v} = \text{MaxEV}(A, \delta)$, and hence

$$\mathcal{E}\langle A\hat{v}, \hat{v} \rangle \geq (1 - \delta)\lambda_{\max}(A).$$

Since $\langle A\hat{v}, \hat{v} \rangle \leq \lambda_{\max}(A)$ and the function $p \mapsto \tau^p$ is monotonically decreasing on $(0, 1]$ for any fixed $\tau \in [0, 1]$, it follows that

$$\mathcal{E} \left[\left(\frac{\langle A\hat{v}, \hat{v} \rangle}{\lambda_{\max}(A)} \right)^p \right] \geq \mathcal{E} \frac{\langle A\hat{v}, \hat{v} \rangle}{\lambda_{\max}(A)} \geq 1 - \delta.$$

Rearranging, we obtain inequality (3.1).

If $p > 1$, then $\hat{v} = \text{MaxEV}(A, \delta_p)$ for $\delta_p := 1 - (1 - \delta)^{1/p}$, which means that

$$\mathcal{E} \langle A\hat{v}, \hat{v} \rangle \geq (1 - \delta_p) \lambda_{\max}(A) = (1 - \delta)^{1/p} \lambda_{\max}(A).$$

Applying now Jensen's inequality (using the fact that $t \mapsto t^p$ is a convex and monotonically increasing function on $\mathbb{R}_+$), we obtain

$$\mathcal{E} [\langle A\hat{v}, \hat{v} \rangle^p] \geq [\mathcal{E} \langle A\hat{v}, \hat{v} \rangle]^p \geq [(1 - \delta)^{1/p} \lambda_{\max}(A)]^p = (1 - \delta) [\lambda_{\max}(A)]^p. \quad \square$$

Definition 3.4 (Relatively inexact maximal left/right singular vectors). Let $A \in \mathbb{R}^{m \times n}$ be a matrix and $\delta \in (0, 1)$ be a scalar. A random vector $\hat{u} \in \mathbb{R}^m$ is called a δ -relatively inexact stochastic maximal left singular vector of A iff $\|\hat{u}\| = 1$ (a.s.) and

$$\mathcal{E} \|A^T \hat{u}\| \geq (1 - \delta) \sigma_{\max}(A).$$

A *relatively inexact stochastic maximal left singular vector oracle* is a procedure MaxLeftSV that takes a matrix $A \in \mathbb{R}^{n \times m}$ and a scalar $\delta \in (0, 1)$, and returns a random vector $\hat{u} = \text{MaxLeftSV}(A, \delta)$ such that $\hat{u}$ is a δ -relatively inexact stochastic maximal left singular vector.

Similarly, a random vector $\hat{v} \in \mathbb{R}^n$ is called a δ -relatively inexact maximal right singular vector of A iff $\|\hat{v}\| = 1$ (a.s.) and

$$\mathcal{E} \|A\hat{v}\| \geq (1 - \delta) \sigma_{\max}(A).$$

The maximal left/right singular vector oracles can be implemented in terms of the maximal eigenvector oracle of degree $1/2$.

Algorithm 3.2		
Signature:	$\text{MaxLeftSV} =$	
	$\text{MakeMaxLeftSV}(\text{MaxEV}_{1/2})$	
Input: $\text{MaxEV}_{1/2}$ —relatively inexact maximal eigenvector oracle of degree $1/2$.		
Output: MaxLeftSV —relatively inexact maximal left singular vector oracle.		
Return the function $(A, \delta) \mapsto \text{MaxEV}_{1/2}(AA^T, \delta)$.		

Algorithm 3.3		
Signature:	$\text{MaxRightSV} =$	
	$\text{MakeMaxRightSV}(\text{MaxEV}_{1/2})$	
Input: $\text{MaxEV}_{1/2}$ —relatively inexact maximal eigenvector oracle of degree $1/2$.		
Output: MaxRightSV —relatively inexact maximal right singular vector oracle.		
Return the function $(A, \delta) \mapsto \text{MaxEV}_{1/2}(A^T A, \delta)$.		

Lemma 3.5. *Algorithms 3.2 and 3.3 indeed construct relatively inexact maximal left/right singular vector oracles, respectively.*

Proof. Let $A \in \mathbb{R}^{m \times n}$ and $\delta \in (0, 1)$ be arbitrary, and let $\hat{u} := \text{MaxLeftSV}(A, \delta)$ be the output of the MaxLeftSV procedure constructed by Algorithm 3.2. By our definitions, $\hat{u} = \text{MaxEV}_{1/2}(AA^T, \delta)$. Since $\text{MaxEV}_{1/2}$ is a relatively inexact stochastic maximal eigenvector oracle of degree $1/2$, this means that $\|\hat{u}\| = 1$ (a.s.) and

$$\mathcal{E}\|A^T \hat{u}\| = \mathcal{E}[\langle AA^T \hat{u}, \hat{u} \rangle^{1/2}] \geq (1 - \delta)[\lambda_{\max}(AA^T)]^{1/2} = (1 - \delta)\sigma_{\max}(A).$$

This proves that $\hat{u}$ is a δ -relatively inexact stochastic maximal left singular vector of A .

Thus, Algorithm 3.2 is indeed correct. The justification for Algorithm 3.3 is completely similar. $\square$

Definition 3.6 (Relatively inexact maximal pair of singular vectors). Given a matrix $A \in \mathbb{R}^{m \times n}$, a scalar $\delta \in (0, 1)$, and random vectors $\hat{u} \in \mathbb{R}^m$ and $\hat{v} \in \mathbb{R}^n$, we say that $(\hat{u}, \hat{v})$ is a δ -relatively inexact stochastic maximal pair of singular vectors of A iff $\|\hat{u}\| = \|\hat{v}\| = 1$ (a.s.) and

$$\mathcal{E}\langle A\hat{v}, \hat{u} \rangle \geq (1 - \delta)\sigma_{\max}(A).$$

A *relatively inexact stochastic maximal pair-of-singular-vectors oracle* is a procedure MaxSVPair that accepts a matrix $A \in \mathbb{R}^{m \times n}$ and a scalar $\delta \in (0, 1)$, and returns a pair of random vectors $(\hat{u}, \hat{v}) = \text{MaxSVPair}(A, \delta)$ such that $(\hat{u}, \hat{v})$ is a δ -relatively inexact stochastic maximal singular vector pair.

To construct a pair of maximal singular vectors, we may first compute a left singular vector and then construct the right one from it. Alternatively, we can compute the right singular vector and then construct the left one from it.

Algorithm 3.4	Algorithm 3.5
Signature: MaxSVPair = MakeMaxSVPairFromLeft(MaxLeftSV)	Signature: MaxSVPair = MakeMaxSVPairFromRight(MaxRightSV)
Input: MaxLeftSV—relatively inexact maximal left singular vector oracle.	Input: MaxRightSV—relatively inexact maximal right singular vector oracle.
Output: MaxSVPair—relatively inexact maximal pair-of-singular-vectors oracle.	Output: MaxSVPair—relatively inexact maximal pair-of-singular-vectors oracle.
1. Return the function MaxSVPair = MaxSVPair(A, δ) defined by the following computation: a) $\hat{u} := \text{MaxLeftSV}(A, \delta)$. b) $\hat{v}' := A^T \hat{u}$. c) $\hat{v} := \frac{\hat{v}'}{\ \hat{v}'\ }$. d) Return $(\hat{u}, \hat{v})$.	1. Return the function MaxSVPair = MaxSVPair(A, δ) defined by the following computation: a) $\hat{v} := \text{MaxRightSV}(A, \delta)$. b) $\hat{u}' := A\hat{v}$. c) $\hat{u} := \frac{\hat{u}'}{\ \hat{u}'\ }$. d) Return $(\hat{u}, \hat{v})$.

Lemma 3.7. *Both Algorithms 3.4 and 3.5 are correct in the sense that their output is indeed a relatively inexact stochastic maximal pair-of-singular-vectors oracle.*

Proof. We only prove the claim for Algorithm 3.4, as the proof for Algorithm 3.5 is completely similar.

Let $A \in \mathbb{R}^{m \times n}$ and $\delta \in (0, 1)$ be arbitrary, and let $(\hat{u}, \hat{v}) = \text{MaxSVPair}(A, \delta)$ be the output of the MaxSVPair procedure constructed by Algorithm 3.4. From our definition of $\hat{u}$ and the fact that MaxLeftSV is a relatively inexact maximal left singular vector oracle, it follows that $\|\hat{u}\| = 1$ (a.s.) and

$$\mathcal{E}\|A^T \hat{u}\| \geq (1 - \delta)\sigma_{\max}(A).$$

On the other hand, from the definitions of $\hat{v}'$ and $\hat{v}$, it is clear that $\|\hat{v}\| = 1$ and

$$\langle A\hat{v}, \hat{u} \rangle = \langle \hat{v}, \hat{v}' \rangle = \|\hat{v}'\| = \|A^T \hat{u}\|.$$

Combining the above two displays, we conclude that $(\hat{u}, \hat{v})$ is a δ -relatively inexact stochastic maximal pair of singular vectors of A . $\square$

3.1.2 Power Method

The simplest oracle for $\text{MaxEV}(A, \delta)$ is given by the *Power method*. The idea is to choose a random starting vector $\xi \in \mathbb{R}^n$ and then compute

$$\hat{u}_p := \frac{A^p \xi}{\|A^p \xi\|}$$

for a sufficiently large integer power $p \geq 1$.

For numerical stability, this algorithm is typically implemented in a slightly different form where the normalization is performed after each matrix-vector multiplication.

Algorithm 3.6: Power Method
Signature: $\hat{u} = \text{PowerMethod}(A, p)$.
Input: Matrix $A \in \mathbb{S}_+^n$, number of matrix-vector products p [integer ≥ 1].
Output: Random vector $\hat{u} \in \mathcal{S}^{n-1}$.
1. Sample $u_0 \sim \text{Unif}(\mathcal{S}^{n-1})$. 2. Iterate for $k = 1, \dots, p$: a) $\hat{u}'_k := A\hat{u}_{k-1}$. b) $\hat{u}_k := \frac{\hat{u}'_k}{\ \hat{u}'_k\ }$. 3. Return $\hat{u}_p$.

The standard convergence result about the Power Method is as follows.

Theorem 3.8 (Theorem 3.1 in [3]). *Suppose that the input in Algorithm 3.6 is such that $n \geq 8$ and $p \geq 2$. Then, the output vector $\hat{u}$ is a δ_p -relatively inexact stochastic unit eigenvector of A , where*

$$\delta_p := 0.871 \frac{\ln n}{p}.$$

From the above theorem, we know how to, given an accuracy $\delta \in (0, 1)$, choose the number of iterations p sufficient to guarantee that the output of the Power method is a δ -relatively inexact stochastic unit eigenvector of A . This allows us to build an oracle for $\text{MaxEV}(A, \delta)$.

Algorithm 3.7: Power Oracle for Maximal Eigenvector
Signature: $\hat{u} = \text{MaxEVPower}(A, \delta)$.
Input: Matrix $A \in \mathbb{S}_+^n$, accuracy $\delta \in (0, 1)$. Output: Random vector $\hat{u} \in \mathbb{R}^n$ such that $\hat{u}$ is a δ -relatively inexact stochastic maximal eigenvector of A .
1. Set $p := \lceil 0.871 \frac{\ln n}{\delta} \rceil$. 2. Return $\text{PowerMethod}(A, p)$.

Theorem 3.9. *For any $n \geq 8$, Algorithm 3.7 indeed returns a δ -relatively inexact stochastic maximal eigenvector of A . The running time of the method is*

$$\left\lceil \frac{\ln n}{\delta} \right\rceil \times \text{Time}(\text{MatVec}(A)) + O\left(\frac{n \ln n}{\delta}\right),$$

where $\text{Time}(\text{MatVec}(A))$ is the running time of a matrix-vector multiplication for A .

Proof. The fact that $\hat{u}$ is indeed a δ -relatively inexact stochastic maximal eigenvector of A follows from Theorem 3.8 (note that, since $n \geq 8$ and $\delta \in (0, 1)$, we have $p \geq \lceil 0.871 \cdot \ln 8 \rceil = \lceil 1.81 \dots \rceil = 2$).

To establish the complexity bound, note that, at each iteration of the loop, only one matrix-vector multiplication is performed. Therefore, the total number of matrix vector multiplications is

$$p = \left\lceil 0.871 \frac{\ln n}{\delta} \right\rceil \leq \left\lceil \frac{\ln n}{\delta} \right\rceil.$$

The number of auxiliary operations at each iteration is $O(n)$. Therefore, the extra running time complexity is

$$O(np) \leq O\left(n\left(\frac{\ln n}{\delta} + 1\right)\right) = O\left(\frac{n \ln n}{\delta}\right)$$

since $\frac{\ln n}{\delta} \geq 1$ for $n \geq 8$. □

3.1.3 Lanczos Algorithm

The Lanczos method chooses a random vector $\xi \in \mathcal{S}^{n-1}$ and then searches for a maximizer of the Rayleigh quotient in the p -th Krylov subspace generated by ξ :

$$\hat{v}_p \in \operatorname{Argmax}\{\langle Av, v \rangle : v \in \operatorname{Klv}_p(A, \xi) \cap \mathcal{S}^{n-1}\},$$

where

$$\operatorname{Klv}_p(A, \xi) := \operatorname{span}\{\xi, A\xi, \dots, A^p\xi\}.$$

We refer to any vector $\hat{v}_p$ satisfying the above inclusion as a *Lanczos maximal eigenvector* of order p for (A, ξ) . Such a vector can be found very efficiently by using only $O(p)$ matrix-vector products with matrix A .

The key step is to first find an orthonormal basis for the Krylov subspace $\operatorname{Klv}_p(A, \xi)$ —a matrix $Q \in \mathbb{R}^{n \times (p+1)}$ with $Q^T Q = I$ (identity matrix) and $\operatorname{im} Q = \operatorname{Klv}_p(A, \xi)$ —in which A is tridiagonal:

$$Q^T A Q = \operatorname{Tridiag}(\alpha, \beta),$$

where $\operatorname{Tridiag}(\alpha, \beta)$ the $(p+1) \times (p+1)$ symmetric tridiagonal matrix with some (known) vector $\alpha \in \mathbb{R}^{p+1}$ on the main diagonal and some (known) vector $\beta \in \mathbb{R}^p$ on the sub- and super diagonals. In what follows, we refer to such an (α, β, Q) as a *Lanczos tridiagonalization triple* of order p for (A, ξ) . This triple can be constructed using the following algorithm.

Algorithm 3.8: Lanczos Tridiagonalization	
Signature: $(\alpha, \beta, Q) = \operatorname{LanczosTridiag}(A, \xi, p)$.	
Input: Matrix $A \in \mathbb{S}^n$, vector $\xi \in \mathcal{S}^{n-1}$, number of iterations $p \geq 1$.	
Output: Vectors $\alpha \in \mathbb{R}^{p+1}$, $\beta \in \mathbb{R}^p$, and matrix $Q \in \mathbb{R}^{n \times (p+1)}$ such that (α, β, Q) is a Lanczos tridiagonalization triple of order p for (A, ξ) .	
1. Set $q_0 := \xi$. Compute $q'_0 := Aq_0$, $\alpha_0 := \langle q'_0, q_0 \rangle$, and $r_0 := q'_0 - \alpha_0 q_0$. 2. Iterate for $k = 0, \dots, p-1$: a) Compute $\beta_k := \ r_k\$, $q_{k+1} := r_k / \beta_k$, and $q'_{k+1} := Aq_{k+1}$. b) Compute $\alpha_{k+1} := \langle q'_{k+1}, q_{k+1} \rangle$ and $r_{k+1} := q'_{k+1} - \alpha_{k+1} q_{k+1} - \beta_k q_k$. 3. Return $\alpha := (\alpha_0, \dots, \alpha_p)$, $\beta := (\beta_0, \dots, \beta_{p-1})$, $Q := [q_0, \dots, q_p]$.	

Theorem 3.10 (Theorem 10.1.1 in [1]). *Algorithm 3.8 is correct in the sense that it indeed produces a Lanczos tridiagonalization triple.*

Combining Algorithm 3.8 with any *exact* algorithm $\operatorname{MaxEVOFTridiag}$ for computing a maximal unit eigenvector of a tridiagonal matrix, we get the Lanczos method for computing a leading eigenvector of A .

Algorithm 3.9: Lanczos Algorithm
Signature: $\hat{v} = \text{LanczosAlg}(A, p)$.
Input: Matrix $A \in \mathbb{S}^n$, number of iterations $p \geq 1$.
Output: $\hat{v} \in \mathcal{S}^{n-1}$ —Lanczos maximal eigenvector of order p for (A, ξ) for a randomly sampled $\xi \sim \text{Unif}(\mathcal{S}^{n-1})$.
1. Sample $\xi \sim \text{Unif}(\mathcal{S}^{n-1})$. 2. Compute $(\alpha, \beta, Q) := \text{LanczosTridiag}(A, \xi, p)$. 3. Compute $\hat{x} := \text{MaxEVOOfTridiag}(\alpha, \beta)$. 4. Return $\hat{v} := Q\hat{x}$.

In principle, we can use any algorithm in place of MaxEVOOfTridiag . However, for concreteness, we will assume that this is the standard QR decomposition-based algorithm for computing eigenvectors and eigenvalues of a symmetric tridiagonal matrix (see Section 8.3 in [1]). The complexity of such an algorithm is $O(p^2)$, where p is the dimension of the tridiagonal matrix. Even if the original dimension n of the matrix was huge, the value of p is typically rather small, so this complexity is affordable.

Theorem 3.11. *The vector $\hat{v}$ returned by Algorithm 3.9 is indeed a Lanczos maximal eigenvector of order p for (A, ξ) .*

Proof. According to the guarantees of Algorithm 3.8, at the end of Step 2 of Algorithm 3.9, the matrix Q is such that its columns form an orthonormal basis for $\text{Klv}_p(A, \xi)$ and $Q^T A Q = T$, where $T := \text{Tridiag}(\alpha, \beta)$. Thus, any vector v from $\text{Klv}_p(A, \xi) \cap \mathcal{S}^{n-1}$ can be (uniquely) parameterized as $v = Qx$, where $x \in \mathcal{S}^p$. Consequently, v is a Lanczos maximal eigenvector of order p for (A, ξ) iff $v = Qx$, where x maximizes

$$\langle A(Qx), Qx \rangle = \langle Q^T A Q x, x \rangle = \langle T x, x \rangle$$

over all $x \in \mathcal{S}^p$, or, equivalently, iff x is a maximal unit eigenvector of T . The claim now follows from the fact that $\hat{x}$ obtained at Step 3 is indeed a maximal unit eigenvector of T and $\hat{v} = Q\hat{x}$. $\square$

The classical convergence bound for the Lanczos algorithm is as follows.

Theorem 3.12 (Theorem 3.2 in [3]). *Suppose that the input in Algorithm 3.9 is such that A is positive semidefinite, $n \geq 8$ and $p \geq 3$. Then, the output $\hat{v}$ generated by the method is a δ_p -relatively inexact stochastic unit eigenvector of A , where*

$$\delta_p := 2.575 \left(\frac{\ln n}{p} \right)^2.$$

Using the previous bound, we can now select p for any required δ and thus construct the Lanczos oracle for $\text{MaxEV}(A, \delta)$.

Algorithm 3.10: Lanczos Oracle for Computing Maximal Eigenvector
Signature: $\hat{v} = \text{MaxEVLanczos}(A, \delta)$.
Input: Matrix $A \in \mathbb{S}^n$, accuracy $\delta \in (0, 1)$.
Output: $\hat{v} \in \mathcal{S}^{n-1}$ — δ -relatively inexact stochastic unit eigenvector of A .
1. Compute $p := \lceil 1.605 \frac{\ln n}{\sqrt{\delta}} \rceil$. 2. Return $\text{LanczosAlg}(A, p)$.

Theorem 3.13. *The output of Algorithm 3.10 is indeed a δ -relatively inexact stochastic unit eigenvector of A . The total running time of the algorithm is*

$$\left(2 \frac{\ln n}{\sqrt{\delta}} + 1\right) \times \text{Time}(\text{MatVec}(A)) + O\left(\frac{n \ln n}{\sqrt{\delta}}\right).$$

Proof. The fact that $\hat{v}$ is indeed a δ -relatively inexact stochastic unit eigenvector of A follows immediately from Theorem 3.12 and the fact that $\sqrt{2.575} = 1.6046 \dots \leq 1.605$.

To justify the time complexity, note that all matrix-vector products with A are performed only inside the call to $\text{LanczosTridiag}(A, p)$ (one multiplication at each iteration plus one extra during initialization). The extra complexity at each iteration inside LanczosTridiag is $O(n)$, and $O(n)$ during initialization. The total extra complexity of the call to $\text{LanczosTridiag}(A, p)$ is thus $O(np)$.

Further, the complexity of $\text{MaxEVOFTridiag}(\alpha, \beta)$ is $O(p^2) = O(np)$ since $p \leq n$ (otherwise, the Lanczos algorithm terminates). $\square$

3.2 Maximal Eigenvalue of Symmetric Matrix

Lemma 3.14. *Consider the function $f: \mathbb{S}^n \rightarrow \mathbb{R}$ defined by*

$$f(X) := \lambda_{\max}(X).$$

Let $\delta \in (0, 1)$. Consider the oracle $\hat{g}$ defined at each point $X \in \mathbb{S}^n$ by

$$\hat{g}(X) := \hat{v}\hat{v}^T, \quad \hat{v} := \text{MaxEV}(X, \delta).$$

Then, $\hat{g}$ is a δ -relatively inexact stochastic oracle for f .

Proof. Let $X, Y \in \mathbb{S}^n$. Since $\hat{v}$ is a unit vector (a.s.), we have (a.s.)

$$f(Y) = \lambda_{\max}(Y) \geq \langle Y\hat{v}, \hat{v} \rangle = \langle X\hat{v}, \hat{v} \rangle + \langle (Y - X)\hat{v}, \hat{v} \rangle = \langle X\hat{v}, \hat{v} \rangle + \langle \hat{g}(X), Y - X \rangle.$$

Taking now expectations and using the fact that $\hat{v}$ is a δ -relatively inexact eigenvector of X , we get

$$f(Y) = \mathcal{E}f(Y) \geq \mathcal{E}\langle X\hat{v}, \hat{v} \rangle + \langle \mathcal{E}\hat{g}(X), Y - X \rangle \geq (1 - \delta)f(X) + \langle \mathcal{E}\hat{g}(X), Y - X \rangle. \quad \square$$

Sometimes, we need to compute the oracle for the composition of $\lambda_{\max}$ with an affine mapping:

$$f(x) = \lambda_{\max}(Ax + C), \quad x \in \mathbb{R}^d$$

where $A: \mathbb{R}^d \rightarrow \mathbb{S}^n$ is the linear operator

$$Ax := \sum_{i=1}^d x_i A_i,$$

and $A_1, \dots, A_d, C \in \mathbb{S}^n$. In this case, our oracle is

$$\hat{g}(X) = A^* \hat{G}(Ax + C),$$

where $\hat{G}$ is the standard oracle for $\lambda_{\max}$:

$$\hat{G}(Y) = \hat{u} \hat{u}^T, \quad \hat{u} := \text{MaxEV}(Y, \delta),$$

and $A^*: \mathbb{S}^n \rightarrow \mathbb{R}^d$ is the adjoint operator

$$A^*G = (\langle A_i, G \rangle)_{i=1}^d.$$

In this case, we can evaluate $\hat{g}(x)$ without forming any intermediate matrices:

$$\hat{g}(x) = (\langle A_i \hat{u}, \hat{u} \rangle)_{i=1}^d, \quad \hat{u} := \text{MaxEV}(Ax + C, \delta).$$

The complexity of this operation is just extra d matrix-vector multiplications of A_i by $\hat{u}$.

3.3 Maximal Singular Value

Lemma 3.15. *Consider the function $f: \mathbb{R}^{m \times n} \rightarrow \mathbb{R}$ defined by*

$$f(X) := \sigma_{\max}(X).$$

Let $\delta \in (0, 1)$. Consider the oracle $\hat{g}$ defined at each point $X \in \mathbb{R}^{m \times n}$ by

$$\hat{g}(X) := \hat{u} \hat{v}^T, \quad (\hat{u}, \hat{v}) := \text{MaxSVPair}(X, \delta).$$

Then, $\hat{g}$ is a δ -relatively inexact stochastic oracle for f .

Proof. Let $X, Y \in \mathbb{R}^{m \times n}$. Since $\hat{u}$ and $\hat{v}$ are unit vectors (a.s.), we have (a.s.)

$$f(Y) = \sigma_{\max}(Y) \geq \langle Y \hat{v}, \hat{u} \rangle = \langle X \hat{v}, \hat{u} \rangle + \langle (Y - X) \hat{v}, \hat{u} \rangle = \langle X \hat{v}, \hat{u} \rangle + \langle \hat{g}(X), Y - X \rangle.$$

Taking now expectations and using the fact that $(\hat{u}, \hat{v})$ is a δ -relatively inexact pair of singular vectors of X , we get

$$f(Y) = \mathcal{E}f(Y) \geq \mathcal{E}\langle X \hat{v}, \hat{u} \rangle + \mathcal{E}\langle \hat{g}(X), Y - X \rangle \geq (1 - \delta)f(X) + \mathcal{E}\langle \hat{g}(X), Y - X \rangle. \quad \square$$

3.4 Squared Spectral Norm

Lemma 3.16. Consider the function $f: \mathbb{R}^{m \times n} \rightarrow \mathbb{R}$ defined by

$$f(X) := \|X\|_\infty^2 = [\sigma_{\max}(X)]^2.$$

Let $\delta \in (0, 1)$. Consider the oracle $\hat{g}$ defined at each point $X \in \mathbb{R}^{m \times n}$ by

$$\hat{g}(X) := 2\hat{u}\hat{u}^T X, \quad \hat{u} := \text{MaxEV}(XX^T, \delta).$$

Then, $\hat{g}$ is a δ -relatively inexact stochastic oracle for f .

Proof. Let $X, Y \in \mathbb{R}^{n \times m}$, and let $\hat{u}$ be the random vector from the definition of $\hat{g}(X)$. Since $\hat{u}$ is the output of a relatively inexact maximal eigenvector oracle MaxEV , we have $\|\hat{u}\| = 1$ (a.s.). Consequently (a.s.),

$$f(Y) = [\sigma_{\max}(Y)]^2 = \lambda_{\max}(YY^T) \geq \langle YY^T \hat{u}, \hat{u} \rangle = \langle XX^T \hat{u}, \hat{u} \rangle + \langle (YY^T - XX^T) \hat{u}, \hat{u} \rangle.$$

Note that

$$\begin{aligned} YY^T - XX^T &= (Y - X)Y^T + X(Y - X)^T \\ &= (Y - X)X^T + X(Y - X)^T + (Y - X)(Y - X)^T. \end{aligned}$$

Hence (a.s.),

$$\begin{aligned} f(Y) &\geq \langle XX^T \hat{u}, \hat{u} \rangle + 2\langle (Y - X)X^T \hat{u}, \hat{u} \rangle + \|(Y - X)^T \hat{u}\|^2 \\ &\geq \langle XX^T \hat{u}, \hat{u} \rangle + \langle \hat{g}(X), Y - X \rangle. \end{aligned}$$

Taking now the expectation on both sides and using the fact that $\hat{u}$ is a δ -relatively inexact stochastic eigenvector of XX^T , we obtain

$$\begin{aligned} f(Y) &= \mathcal{E}f(Y) \geq (1 - \delta)\lambda_{\max}(XX^T) + \langle \mathcal{E}\hat{g}(X), Y - X \rangle \\ &= (1 - \delta)f(X) + \langle \mathcal{E}\hat{g}(X), Y - X \rangle. \end{aligned}$$

□

Note that the above oracle is relatively bounded w.r.t. the function.

Lemma 3.17. The oracle $\hat{g}$ from Lemma 3.16 is 2-relatively bounded (in the standard Frobenius norm) w.r.t. the function $\|\cdot\|_\infty^2$.

Proof. Let $X \in \mathbb{R}^{m \times n}$, and let $\hat{u}$ be the random vector from the definition of $\hat{g}(X)$. Since $\|\hat{u}\| = 1$ (a.s.), we have (a.s.)

$$\begin{aligned} \|\hat{g}(X)\|_F^2 &= 4\langle \hat{u}\hat{u}^T X, \hat{u}\hat{u}^T X \rangle = 4\langle XX^T \hat{u}, \hat{u} \rangle \\ &\leq 4\lambda_{\max}(XX^T) = 4[\sigma_{\max}(X)]^2 = 4f(X). \end{aligned}$$

Consequently, $\mathcal{E}[\|\hat{g}(X)\|_F^2] \leq 4f(X)$.

□

4 Spectral Linear Regression

Consider the problem of linear approximation of a given matrix $C \in \mathbb{R}^{n \times m}$ by a given collection of matrices $A_1, \dots, A_d \in \mathbb{R}^{n \times m}$ w.r.t. the matrix infinity norm:

$$f^* := \min_{x \in \mathbb{R}^d} f(x), \quad f(x) := \left\| \sum_{i=1}^d x_i A_i - C \right\|_{\infty}. \quad (4.1)$$

Note that problem (4.1) is very similar to a classical linear regression problem. The only difference is that we measure the residual between matrices in the *spectral* norm instead of the Frobenius one. In view of this analogy, we refer to problem (4.1) as a *spectral linear regression* problem.

In what follows, without loss of generality, we assume that $n \leq m$ (otherwise, we can simply transpose all matrices).

We are going to find an approximate solution to problem (4.1) in *relative scale*. For this, however, it will be convenient to first transform this problem into an equivalent one by squaring the objective function:

$$(f^*)^2 = \min_{x \in \mathbb{R}^d} f^2(x), \quad f^2(x) = \|Ax - C\|_{\infty}^2, \quad (4.2)$$

where $A: \mathbb{R}^d \rightarrow \mathbb{R}^{n \times m}$ is the linear operator

$$Ax := \sum_{i=1}^d x_i A_i, \quad x \in \mathbb{R}^d. \quad (4.3)$$

Let us show that problem (4.2) fits the setting from Section 2.3.

First, note that Assumption 2.8 is satisfied as problem (4.2) is unconstrained.

Further, let us equip the space $\mathbb{R}^{n \times m}$ with the standard Frobenius norm:

$$\|X\| := \|X\|_F, \quad X \in \mathbb{R}^{n \times m}.$$

In the notation of Section 2.3, this is the Euclidean seminorm $\|\cdot\|_{B_1}$ with $B_1 = I$ (identity operator in $\mathbb{R}^{n \times m}$).

Clearly, we have

$$f^2(x) = F(Ax - C), \quad \forall x \in \mathbb{R}^d,$$

where $F: \mathbb{R}^{n \times m} \rightarrow \mathbb{R}$ is the squared spectral norm:

$$F(Y) := \|Y\|_{\infty}^2.$$

Note that, for any $X \in \mathbb{R}^{n \times m}$, we have $F(Y) = [\sigma_{\max}(X)]^2 \geq \frac{1}{n} \|X\|_F^2$ (recall that $n \leq m$). Thus, the function F is consistent with the norm $\|\cdot\|$ with the following parameters:

$$\gamma_0 := \frac{1}{n}, \quad \hat{Y}_0 := 0. \quad (4.4)$$

From Lemma 2.10, it follows that, w.r.t. the seminorm $\|\cdot\|_B$ induced by the Gram matrix

$$B = A^* A = (\langle A_i, A_j \rangle)_{i,j=1}^d, \quad (4.5)$$

the function f is also consistent with parameters γ_0 and

$$\hat{x}_0 = T(0, -A^*C),$$

where $T(\cdot, \cdot)$ is the gradient step operation:

$$T(\bar{x}, g) := \operatorname{argmin}_{x \in \mathbb{R}^d} \left\{ \langle g, x \rangle + \frac{1}{2} \|x - \bar{x}\|_B^2 \right\}, \quad \bar{x} \in \mathbb{R}^d, \quad g \in (\ker B)^\perp.$$

Note that the point $T := T(\bar{x}, g)$ can be computed by solving the following linear system (which is guaranteed to be solvable):

$$B(T - \bar{x}) = -g.$$

It remains to specify an (efficiently computable) relatively inexact stochastic oracle for the function F . According to our discussion in Sections 2.3 and 3.4, a good choice is the oracle $\hat{g}(x) = A^* \hat{G}(Ax - C)$, where $\hat{G}$ is the standard $(\Delta/2)$ -relatively inexact oracle for the squared spectral norm from Lemma 3.16 (induced by our choice of a relatively inexact stochastic maximal eigenvector oracle MaxEV), and $\Delta \in (0, 1)$ is a fixed constant (to be specified later).

Recall from Lemma 3.17 that the oracle $\hat{G}$ is 2-relatively bounded (in the standard Frobenius norm) w.r.t. the function F . Therefore, according to Lemma 2.10, the oracle $\hat{g}$ is also 2-relatively bounded but w.r.t. the function f and in the seminorm $\|\cdot\|_B$ induced by the Gram matrix (4.5).

Applying now Algorithm 2.1 with the oracle $\hat{g}$, initial point $x_0 = \hat{x}_0$ constant $L = 2$, accuracy $\delta' = \Delta/2$ and step sizes

$$a_k = \frac{L}{2\delta'}, \quad k \geq 0, \tag{4.6}$$

we conclude, from Theorem 2.5, that, once the number of iterations N performed by the algorithm becomes sufficiently large, namely,

$$N \geq \frac{2L}{\gamma_0(\delta')^2} = \frac{16}{\gamma_0\Delta^2} = \frac{16n}{\Delta^2} \tag{4.7}$$

(see (4.4)), the point $x_N \in \mathbb{R}^d$ constructed by the algorithm is guaranteed to be a Δ -relatively inexact solution to problem (4.2):

$$(1 - \Delta)\mathcal{E}[f^2(x_N)] \leq (f^*)^2. \tag{4.8}$$

Recall, however, that our initial problem was (4.1), not (4.2). Let us therefore see what guarantees we have for the point x_N in terms of our initial problem. Using Jensen's inequality in (4.8), we get

$$\sqrt{1 - \Delta} \mathcal{E}f(x_N) \leq \sqrt{(1 - \Delta)\mathcal{E}[f^2(x_N)]} \leq f^*.$$

Hence, for any given $\delta \in (0, 1)$, choosing

$$\Delta := 1 - (1 - \delta)^2 = (2 - \delta)\delta \quad (\in (0, 1)), \tag{4.9}$$

we can guarantee that the point x_N is a δ -relatively inexact solution to our original problem (4.1),

$$(1 - \delta)\mathcal{E}f(x_N) \leq f^*,$$

for any $N \geq N(\delta)$, where

$$N(\delta) := \frac{16n}{[(2 - \delta)\delta]^2} \leq \frac{16n}{\delta^2} \quad (4.10)$$

(see (4.7)).

5 Numerical Experiments

Let us present preliminary computational results for our algorithms as applied for solving the spectral linear regression problem (4.1) using the setup from Section 4. We set the target relative accuracy to one percent:

$$\delta := 0.01, \quad (5.1)$$

which is a typical choice in most engineering applications.

To be able to assess the performance of our optimization methods, we generate data for problem (4.1) in a special way. Specifically, we choose the matrix $C \in \mathbb{R}^{n \times m}$ to be diagonal such that its largest element (in absolute value) is fixed and is located in the top left corner:

$$C = \text{Diag}(1, c_2, \dots, c_n), \quad |c_i| \leq 1, \quad 2 \leq i \leq n, \quad (5.2)$$

while the matrices $A_1, \dots, A_d \in \mathbb{R}^{n \times m}$ are constructed in such a way so that each of them has zero in the top left corner:

$$A_i^{(1,1)} = 0, \quad 1 \leq i \leq d. \quad (5.3)$$

This way of generating data ensures that the optimal value for our problem is known (see Lemma A.5):

$$f^* = 1.$$

The other diagonal elements $c_2, \dots, c_n$ of C and all nonzero elements of matrices $A_1, \dots, A_d$ are generated randomly from the standard uniform distribution on the interval $[-1, 1]$.

The data for our experiments is generated to be *sparse*. Specifically, each of the matrices $A_1, \dots, A_d$ contains only $s := 5$ nonzero elements in each column. The s row indices of nonzero elements in each column $1 \leq j \leq m$ are randomly selected (without repetition) from the uniform distribution on the set $\{1, \dots, n\}$ if $j > 1$ and $\{2, \dots, n\}$ if $j = 1$ (so that constraint (5.3) is respected).

The specific values of parameters d , n and m , that we consider in the experiments, are shown in Table 5.1, together with the corresponding theoretical number of iterations $N(\delta)$ that was computed according to (4.10).

In what follows, we present the results in form of convergence plots for our methods. Each curve on such a plot displays the dependence of the relative accuracy $\delta_k \in (0, 1)$ of the current approximate solution x_k constructed by the method against the current iteration

d	n	m	$N(\delta)$
400	100	200	4 040 303
800	200	400	8 080 605
2 000	500	1 000	20 201 511
4 000	1 000	2 000	40 403 021

Table 5.1: Problem instances used in our experiments.

number k (or the total computational time taken by the method up to iteration k). The accuracy δ_k is defined as the smallest number such that $(1 - \delta_k)f(x_k) \leq f^*$, i.e.,

$$\delta_k = 1 - f(x_k)/f^*. \quad (5.4)$$

Note from (4.1) that we cannot compute $f(x_k)$ exactly as it requires computing the largest singular value of the (potentially big) matrix $X_k := Ax_k - C$. Therefore, in practice, we actually approximate it by running the standard Power Method for a sufficiently large number of iterations (until the eigenvalue approximation stabilizes) to compute the largest eigenvalue of the matrix $X_k X_k^T$ and then take the square root. Such an approximation is quite efficient and is sufficiently accurate for any practical purposes.

The code for our experiments is written in C++ and uses the Eigen 3 library [2] for matrix computations. It is compiled and run on a laptop with the Intel Core i7-8650U CPU, 16 GiB RAM, and Ubuntu 22.04 OS using the Clang 14 compiler. For performing linear algebra operations, the Eigen library is allowed to use all 8 available threads.

5.1 Gradient Method vs Dual Averaging

In the first experiment, we compare the Gradient Method (Algorithm 2.1 with constant step sizes (4.6)) against the Dual Averaging method (Algorithm 2.2 with parameters given by (2.22)–(2.24)). Both methods use the same oracle based on the Power algorithm for computing the maximal eigenvector.

The results are shown in Fig. 5.1 for two different instances of our problem. The left column displays the convergence in terms of iteration numbers, while the right columns displays the convergence in terms of the running time.

As we can see, there is a huge difference between the two methods: the Gradient Method with fixed step sizes is significantly slower than the Dual Averaging method with dynamically chosen parameters, both in terms of the iteration number and, especially, the running time (where the difference reaches *several orders of magnitude!*).

Nevertheless, both methods have successfully reached the required target accuracy (5.1) in all cases. What is very interesting, however, is that the actual number of iterations it took for them to do that was much smaller than was predicted by the worst-case theoretical estimate $N(\delta)$ (see Table 5.1): by *two orders of magnitude* for the Gradient Method, and by approximately *three to four orders of magnitude* for Dual Averaging. It is, of course, an interesting open question for future research—to investigate why there is such a huge difference between theory and practice.

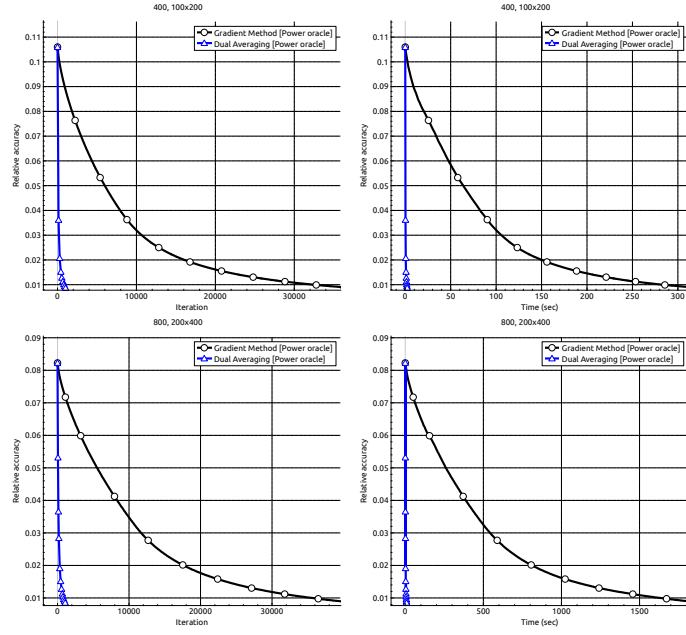


Figure 5.1: Comparison between different methods for the first two problem instances from Table 5.1.

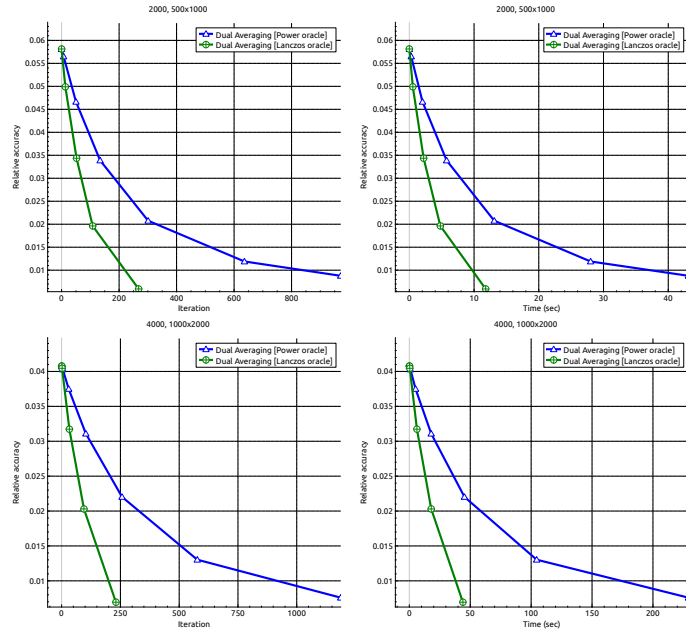


Figure 5.2: Comparison between different oracles for the final two problem instances from Table 5.1.

5.2 Power Oracle vs Lanczos Oracle

As we have seen in Section 5.1, the Gradient Method with fixed step sizes is completely impractical. Let us therefore consider only the Dual Averaging method now but look at the difference between two different oracles: the Power oracle and the Lanczos oracle.

The results are shown in Fig. 5.2, where we now consider much larger problem instances than before. As in Fig. 5.1, the left column displays the convergence in terms of iteration numbers, and the right column—in terms of the running time.

As expected, the method with the Lanczos oracle is faster than the other one: the gap between the two grows with the iteration counter (or the achieved relative accuracy level) and, in the end, reaches, in our particular case, approximately 4 times, both in terms of the iteration number and the running time. Although the difference is not as dramatic as between the two different methods from Section 5.1, it is still quite significant, especially for large-scale problems that require hundreds and thousands of seconds of computations.

A Auxiliary Results

Lemma A.1. *Let $Q \subseteq \mathbb{E}$ be a set, and let $A: \mathbb{E} \rightarrow \mathbb{E}_1$ and $C: \mathbb{E}_1 \rightarrow \mathbb{E}_2$ be linear transformations. Then, the following implication³ holds:*

$$A(Q) + \ker C \text{ is closed} \implies Q + \ker(CA) \text{ is closed.}$$

Proof. Let $A(Q) + \ker C$ be closed, and let $(z_k)_{k=1}^\infty$ be a sequence in $Q + \ker(CA)$ converging to a point $z \in \mathbb{E}$. Let us prove that $z \in Q + \ker(CA)$. Note that, for any $k \geq 1$, we have $Az_k \in A(Q) + A\ker(CA) \in A(Q) + \ker C$. Since A is a continuous mapping (as a linear transformation between finite-dimensional vector spaces) and $z_k \rightarrow z$, it holds that $Az_k \rightarrow Az$. Furthermore, $Az \in A(Q) + \ker C$ since $A(Q) + \ker C$ is a closed set. Thus, $Az = Ax + h$ for some $x \in Q$ and $h \in \ker C$. Consequently, $CA(z - x) = Ch = 0$, which means that $z - x \in \ker(CA)$. But then $z = x + (z - x) \in Q + \ker(CA)$. $\square$

Lemma A.2. *Let $Q \subseteq \mathbb{E}$ be a set, $\mathbb{L} \subseteq \mathbb{E}$ be a linear subspace, $\mathbb{L}^c \subseteq \mathbb{E}$ be a complementary subspace to $\mathbb{L}$, and let $P_{\mathbb{L}^c}: \mathbb{E} \rightarrow \mathbb{L}^c$ be the projector⁴ of $\mathbb{E}$ onto $\mathbb{L}^c$ corresponding to the decomposition $\mathbb{E} = \mathbb{L} \oplus \mathbb{L}^c$. Then,*

$$Q + \mathbb{L} \text{ is closed} \iff P_{\mathbb{L}^c}(Q) \text{ is closed.}$$

Proof. Suppose $Q + \mathbb{L}$ is closed. Let $(u_k)_{k=1}^\infty$ be an arbitrary sequence in $P_{\mathbb{L}^c}(Q)$ converging to a point $u \in \mathbb{E}$. Let us prove that $u \in P_{\mathbb{L}^c}(Q)$. Clearly, $u \in \mathbb{L}^c$ since $P_{\mathbb{L}^c}(Q) \subseteq \mathbb{L}^c$ and $\mathbb{L}^c$ is a closed set (as a linear subspace). On the other hand, since $u_k \in P_{\mathbb{L}^c}(Q)$ for all $k \geq 1$, there exists a sequence $(x_k)_{k=1}^\infty$ in Q such that $u_k = P_{\mathbb{L}^c}x_k$ for all $k \geq 1$. Then, $u_k = x_k - P_{\mathbb{L}}x_k \in Q + \mathbb{L}$ for all $k \geq 1$. Since $Q + \mathbb{L}$ is a closed set and $u_k \rightarrow u$, we have $u \in Q + \mathbb{L}$, i.e., $u = x - h$ for some $x \in Q$ and $h \in \mathbb{L}$. Combining this with the fact that $u \in \mathbb{L}^c$, we conclude that $u = P_{\mathbb{L}^c}u = P_{\mathbb{L}^c}x \in P_{\mathbb{L}^c}(Q)$. This proves the “ $\implies$ ” implication.

The “ $\impliedby$ ” implication follows from Lemma A.1 applied to $A := P_{\mathbb{L}^c}$ and $C := I_{\mathbb{E}}$ (the identity operator in $\mathbb{E}$) as $\ker A = \mathbb{L}$ and $\ker C = \{0\}$. $\square$

Lemma A.3. *Let $\mathbb{L} \subseteq \mathbb{E}$ be a linear subspace, and let $f: \mathbb{E} \rightarrow \mathbb{R}$ be a convex function such that*

$$\partial f(x) \cap \mathbb{L}^\perp \neq \emptyset, \quad \forall x \in \mathbb{E}.$$

Then, f is constant along $\mathbb{L}$:

$$f(x + h) = f(x), \quad \forall x \in \mathbb{E}, \forall h \in \mathbb{L}.$$

Proof. Let $x \in \mathbb{E}$ and $h \in \mathbb{L}$. By our assumption, there is $f'(x) \in \partial f(x) \cap \mathbb{L}^\perp$. Hence,

$$f(x + h) \geq f(x) + \langle f'(x), h \rangle = f(x).$$

Similarly, there exists $f'(x + h) \in \partial f(x + h) \cap \mathbb{L}^\perp$, and hence

$$f(x) \geq f(x + h) + \langle f'(x + h), h \rangle = f(x + h).$$

Thus, $f(x + h) = f(x)$. $\square$

³Hereinafter, $A(Q) := \{Ax : x \in Q\}$ is the image of the set Q under the linear transformation A .

⁴Specifically, if $x = x_{\mathbb{L}} + x_{\mathbb{L}^c}$ is the unique decomposition of $x \in \mathbb{E}$ into the sum of elements from $\mathbb{L}$ and $\mathbb{L}^c$, respectively, then $P_{\mathbb{L}^c}x := x_{\mathbb{L}^c}$.

Lemma A.4. Let $f: \mathbb{E} \rightarrow \mathbb{R}$ be a function, $Q \subseteq \mathbb{E}$ be a nonempty set, $\mathbb{L} \subseteq \mathbb{E}$ be a linear subspace, $\mathbb{L}^c$ be a complementary subspace to $\mathbb{L}$, and let $P_{\mathbb{L}^c}: \mathbb{E} \rightarrow \mathbb{L}^c$ be the projector of $\mathbb{E}$ onto $\mathbb{L}^c$ corresponding to the decomposition $\mathbb{E} = \mathbb{L} \oplus \mathbb{L}^c$. Suppose that:

- (i) f is constant along $\mathbb{L}$, i.e., $f(x+h) = f(x)$ for all $x \in \mathbb{E}$ and all $h \in \mathbb{L}$.
- (ii) $Q + \mathbb{L}$ is a closed set.
- (iii) f is a closed function.
- (iv) f restricted to $P_{\mathbb{L}^c}(Q)$ has bounded sublevel sets⁵.

Then, f has a minimizer on Q .

Proof. In view of assumption (i), we can reduce the problem of minimizing f on Q to that of minimizing f on $P_{\mathbb{L}^c}(Q)$:

$$\begin{aligned} \inf_{x \in Q} f(x) &= \inf_{u \in \mathbb{L}^c, h \in \mathbb{L}} \{f(u+h) : u+h \in Q\} = \inf_{u \in \mathbb{L}^c, h \in \mathbb{L}} \{f(u) : u+h \in Q\} \\ &= \inf_{u \in \mathbb{L}^c} \{f(u) : u+h \in Q \text{ for some } h \in \mathbb{L}\} = \inf_{u \in P_{\mathbb{L}^c}(Q)} f(u). \end{aligned}$$

In particular, if f has a minimizer u^* on $P_{\mathbb{L}^c}(Q)$, then f also has a minimizer on Q , which is given by any $x^* \in Q$ such that $P_{\mathbb{L}^c}x^* = u^*$ (at least one such x^* exists by the definition of $P_{\mathbb{L}^c}(Q)$).

It remains to prove that f has a minimizer on $P_{\mathbb{L}^c}(Q)$. According to assumption (ii) and Lemma A.2, the set $P_{\mathbb{L}^c}(Q)$ is closed. Moreover, it is nonempty since Q is assumed to be nonempty. Let $u_0 \in P_{\mathbb{L}^c}(Q)$ be an arbitrary point. It suffices to show that f has a minimizer on the set $L_0 := \{u \in P_{\mathbb{L}^c}(Q) : f(u) \leq f(u_0)\}$. Clearly, $L_0 \neq \emptyset$ (it contains u_0). Furthermore, L_0 is bounded (by assumption (iv)) and closed as the intersection of two closed sets: $P_{\mathbb{L}^c}(Q)$ and $\{u \in \mathbb{E} : f(u) \leq f(u_0)\}$ (whose closedness follows from assumption (iii)). Thus, L_0 is a nonempty compact set and f is a closed function. Hence, by the Weierstrass extreme value theorem, there indeed exists a minimizer of f on L_0 . $\square$

Lemma A.5. Problem (4.1) with data satisfying requirements (5.2) and (5.3) has an optimal solution $x^* = 0$ and the following optimal value:

$$f^* = f(0) = \|C\|_\infty = 1.$$

Proof. It suffices to show that f has a zero subgradient at $x^* = 0$. Note that, for each $x \in \mathbb{R}^d$, we have $f(x) = F(Ax - C)$, where $F: \mathbb{R}^{n \times m} \rightarrow \mathbb{R}$ is the spectral norm function $F(X) = \|X\|_\infty = \max_{u \in \mathcal{S}^{n-1}, v \in \mathcal{S}^{m-1}} \langle Xv, u \rangle$ and $A: \mathbb{R}^d \rightarrow \mathbb{R}^{n \times m}$ is the linear operator defined in (4.3). By standard calculus rules for subgradients, we know that, for any $F'(-C) \in \partial F(-C)$, we have $A^*F'(-C) \in \partial f(0)$, and, for any $X \in \mathbb{R}^{n \times m}$, we have $F'(X) := u(X)[v(X)]^T \in \partial F(X)$, where $u(X) \in \mathcal{S}^{n-1}$ and $v(X) \in \mathcal{S}^{m-1}$ are such that $\langle Xv(X), u(X) \rangle = F(X)$. According to (5.2), we can take $u(-C) := -e_{1,n}$ and $v(-C) = e_{1,m}$, where $e_{1,n} := (1, 0, \dots, 0) \in \mathbb{R}^n$ and $e_{1,m} := (1, 0, \dots, 0) \in \mathbb{R}^m$. This gives us $F'(-C) = -e_{1,n}e_{1,m}^T$. Consequently, $f'(0) = -A^*(e_{1,n}e_{1,m}^T)$ is the vector with elements $[f'(0)]^{(i)} = -\langle A_i, e_{1,n}e_{1,m}^T \rangle = -A_i^{(1,1)} = 0$ (see (5.3)) for any $1 \leq i \leq d$. $\square$

⁵This means that, for any $\alpha \in \mathbb{R}$, the set $\{u \in P_{\mathbb{L}^c}(Q) : f(u) \leq \alpha\}$ is bounded.

References

- [1] G. H. Golub and C. F. Van Loan. *Matrix Computations*. JHU press, 4th edition, 2013.
- [2] G. Guennebaud, B. Jacob, et al. Eigen v3, 2010. URL: <http://eigen.tuxfamily.org>.
- [3] J. Kuczyński and H. Woźniakowski. Estimating the largest eigenvalue by the power and lanczos algorithms with a random start. *SIAM Journal on Matrix Analysis and Applications*, 13(4):1094–1122, Oct. 1992. DOI: 10.1137/0613066. URL: <https://doi.org/10.1137/0613066>.
- [4] Y. Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120(1):221–259, June 2007. DOI: 10.1007/s10107-007-0149-x. URL: <https://doi.org/10.1007/s10107-007-0149-x>.
- [5] Y. Nesterov. *Lectures on Convex Optimization*, volume 137. Springer, second edition, 2018.
- [6] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. Studies in Applied and Numerical Mathematics. SIAM, 1994.