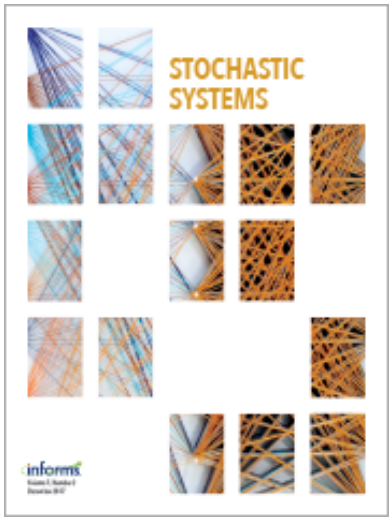
This article was downloaded by: [216.73.217.172] On: 15 November 2026, At: 06:43
Publisher: Institute for Operations Research and the Management Sciences (INFORMS)
INFORMS is located in Maryland, USA



Stochastic Systems

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Subsampling Algorithms for Semidefinite Programming

Alexandre d'Aspremont

To cite this article:
Alexandre d'Aspremont (2011) Subsampling Algorithms for Semidefinite Programming. Stochastic Systems 1(2):274-305. https://doi.org/10.1287/10-SSY018

This work is licensed under a Creative Commons Attribution 4.0 International License. You are free to copy, distribute, transmit and adapt this work, but you must attribute this work as "*Stochastic Systems*. Copyright 2011 The Author(s). https://doi.org/10.1287/10-SSY018, used under a Creative Commons Attribution License: http://creativecommons.org/licenses/by/4.0/."

Copyright © 2011, The author(s)

Please scroll down for article—it is on subsequent pages

With 12,500 members from nearly 90 countries, INFORMS is the largest international association of operations research (O.R.) and analytics professionals and students. INFORMS provides unique networking and learning opportunities for individual professionals, and organizations of all types and sizes, to better understand and use O.R. and analytics tools and methods to transform strategic visions and achieve better outcomes.
For more information on INFORMS, its publications, membership, or meetings visit http://www.informs.org

Stochastic Systems
2011, Vol. 1, No. 2, 274–305
DOI: 10.1214/10-SSY018

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

SUBSAMPLING ALGORITHMS FOR SEMIDEFINITE PROGRAMMING

By Alexandre d'Aspremont

CMAP, École Polytechnique, UMR CNRS 7641

We derive a stochastic gradient algorithm for semidefinite optimization using randomization techniques. The algorithm uses subsampling to reduce the computational cost of each iteration and the subsampling ratio explicitly controls granularity, i.e. the tradeoff between cost per iteration and total number of iterations. Furthermore, the total computational cost is directly proportional to the complexity (i.e. rank) of the solution. We study numerical performance on some large-scale problems arising in statistical learning.

1. Introduction. Beyond classic combinatorial relaxations Goemans and Williamson (1995), semidefinite programming has recently found a new stream of applications in machine learning Lanckriet et al. (2002), geometry Weinberger and Saul (2006), statistics d'Aspremont, Banerjee and El Ghaoui (2006) or graph theory Sun et al. (2006). All these problems have a common characteristic: they have relatively low precision targets but form very large semidefinite programs for which obtaining second order models is numerically hopeless which means that Newton based interior point solvers typically fail before completing even a single iteration. Early efforts focused on exploiting structural properties of the problem (sparsity, block patterns, etc), but this has proven particularly hard for semidefinite programs. For very large problem instances, first-order methods remain at this point the only credible alternative. This follows a more general trend in optimization which seeks to significantly reduce the *granularity* of solvers, i.e. reduce the per iteration complexity of optimization algorithms rather than their total computational cost, thus allowing at least some progress to be made on problems that are beyond the reach of current algorithms.

In this work, we focus on the following spectral norm minimization problem

$$\begin{array}{ll} \text{minimize} & \left\| \sum_{j=1}^{p} y_j A_j + C \right\|_2 - b^T y \\ \text{subject to} & y \in Q, \end{array} \tag{1}$$

Received November 2010.
AMS 2000 subject classifications: Primary 90C22, 90C15.
Keywords and phrases: Semidefinite programming, stochastic optimization, subsampling.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

in the variable $y \in \mathbf{R}^p$, with parameters $A_j \in \mathbf{S}_n$, for $j = 1, \ldots, p$, $b \in \mathbf{R}^p$ and $C \in \mathbf{S}_n$, where Q is a compact convex set. Throughout the paper, we also implicitly assume that the set $Q \subset \mathbf{R}^p$ is simple enough so that the complexity of projecting y on Q is relatively low compared to the other steps in the algorithm.

The idea behind this paper stems from a recent result by Juditsky et al. (2009), who used a mirror descent stochastic approximation algorithm for solving bilinear matrix games (see Nesterov (2009), Polyak and Juditsky (1992) or Nemirovsky and Yudin (1983) for more background), where subsampling is used to perform matrix vector products and produce an approximate gradient. Strikingly, the algorithm has a total complexity of $O(n \log n/\epsilon^2)$, when the problem matrix is $n \times n$, hence only requires access to a negligible proportion of the matrix coefficients as the dimension n tends to infinity. A similar subsampling argument was used in Juditsky, Nemirovskii and Tauvel (2008) to solve a variational inequality representation of maximum eigenvalue minimization problems.

In parallel, recent advances in large deviations and random matrix theory have produced a stream of new randomization results for high dimensional linear algebra (see Achlioptas and McSherry (2007); Drineas, Kannan and Mahoney (2006); Frieze, Kannan and Vempala (2004); Kannan and Vempala (2009) among many others), motivated by the need to perform these operations on very large scale, sometimes streaming, data sets in applications such as machine learning, signal processing, etc. Similar subsampling techniques have been successfully applied to support vector machine classification Kumar, Bhattacharya and Hariharan (2008) or Fourier decomposition. Randomization results were used in Arora and Kale (2007) to produce complexity bounds for certain semidefinite programs arising in combinatorial relaxations of graph problems. Randomization was also used in Burke, Lewis and Overton (2002) and Burke, Lewis and Overton (2005) to approximate subdifferentials of functions that are only differentiable almost everywhere. A randomized algorithm for semidefinite programming based on random walk techniques was also developed in Polyak and Shcherbakov (2007). Finally, a recent stochastic version by Lan (2009) of the algorithm in Nesterov (2007) has the potential to improve the complexity bounds provided by the method in Juditsky et al. (2009).

Our contribution here is to further reduce the granularity of first-order semidefinite programming solvers by combining subsampling procedures with stochastic approximation algorithms to derive stochastic gradient methods for spectral norm minimization with very low complexity per iteration. In practice, significantly larger per iteration complexity and memory require-

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

ments mean that interior point techniques often fail to complete a single iteration on very large problem instances. CPU clock also runs much faster than RAM, so operations small enough to be performed entirely in cache (which runs at full speed) are much faster than those requiring larger data sets. Solver performance on very large problem instances is then often more constrained by memory bandwidth than clock speed, hence everything else being equal, algorithms running many cheap iterations will be much faster than those requiring fewer, more complex ones. Here, subsampling techniques allow us to produce semidefinite optimization algorithms with very low cost per iteration, where all remaining $O(n^2)$ operations have a small constant and can be performed in a single pass over the data.

We also observe that the relative approximation error in computing the spectral norm (or trace norm) of a matrix using subsampling is directly proportional to the numerical rank of that matrix, hence another important consequence of using subsampling techniques to solve large-scale semidefinite programs is that the total complexity of running the algorithm becomes explicitly dependent on the complexity (i.e. rank) of its solution. Most algorithms exploiting the fact that the solution has low rank are not convex (e.g. alternating minimization).

The paper is organized as follows. Section 2 surveys some key results on randomized linear algebra and spectral norm approximations. In Section 3 we then derive a stochastic approximation algorithm for spectral norm minimization with very low cost per iteration and discuss some extensions to statistical learning problems. Finally, we present some numerical experiments in Section 4.

Notation. We write $\mathbf{S}_n$ the set of symmetric matrices of dimension n. For a matrix $X \in \mathbf{R}^{m\times n}$, we write $\|X\|_F$ its Frobenius norm, $\|X\|_{\mathrm{tr}}$ its trace norm, $\|X\|_2$ its spectral norm, $\sigma_i(X)$ its i-th largest singular value and let $\|X\|_\infty = \max_{ij} |X_{ij}|$, while $X^{(i)}$ is the i-th column of the matrix X and $X_{(i)}$ its i-th row. We write $\mathbf{vec}(X)$ the vector of $\mathbf{R}^{mn}$ obtained by stacking up the columns of the matrix X and $\mathbf{NumRank}(X)$ the numerical rank of the matrix X, where $\mathbf{NumRank}(X) = \|X\|_F^2/\|X\|_2^2$. Finally, when $x \in \mathbf{R}^n$ is a vector, we write $\|x\|_2$ its Euclidean norm, while $\|\cdot\|$ is a general norm on $\mathbf{R}^m$ and $\|\cdot\|_*$ its dual norm.

2. Randomized linear algebra. In this section, we survey several results by Drineas, Kannan and Mahoney (2006) which, after a single pass on the data, sample columns to approximate matrix products and produce low rank matrix approximations with a complexity of $O(sn)$ where s is the sampling rate.

Algorithm 1 Matrix multiplication

Input: $A \in \mathbf{R}^{m\times n}$, $B \in \mathbf{R}^{n\times p}$ and s such that $1 \le s \le n$.

1: Define a probability vector $q \in \mathbf{R}^n$ such that

$$q_i = \frac{\|A^{(i)}\|_2 \|B_{(i)}\|_2}{\sum_{j=1}^n \|A^{(j)}\|_2 \|B_{(j)}\|_2}, \quad i = 1, \ldots, n.$$

2: Define subsampled matrices $C \in \mathbf{R}^{m\times s}$ and $R \in \mathbf{R}^{s\times p}$ as follows.
3: **for** $i = 1$ to s **do**
4: Pick $j \in [1, n]$ with $\mathbf{P}(j = l) = q_l$.
5: Set $C^{(i)} = A^{(j)}/\sqrt{sq_j}$ and $R_{(i)} = B_{(j)}/\sqrt{sq_j}$.
6: **end for**

Output: Matrix product CR approximating AB.

2.1. *Randomized matrix multiplication.* By construction, $\mathbf{E}[CR] = AB$, and the following randomization result from Drineas, Kannan and Mahoney (2007) controls the precision of the approximations in algorithm 1.

LEMMA 1. *Let $A \in \mathbf{R}^{m\times n}$, $B \in \mathbf{R}^{n\times p}$, given a subsampling rate s such that $1 \le s \le n$, suppose that $C \in \mathbf{R}^{m\times s}$ and $R \in \mathbf{R}^{s\times p}$ are computed according to algorithm 1 above, then*

$$\mathbf{E}[\|AB - CR\|_F^2] \le \frac{1}{s}\|A\|_F^2 \|B\|_F^2$$

and if $\beta \in [0, 1]$ with $\eta = 1 + \sqrt{8\log(1/\beta)}$ then

$$\|AB - CR\|_F^2 \le \frac{\eta^2}{s}\|A\|_F^2 \|B\|_F^2$$

with probability at least $1 - \beta$.

PROOF. See Theorem 1 in Drineas, Kannan and Mahoney (2007). □

Note that using the adaptive probabilities q_i is crucial here. The error bounds increase by a factor n when $q_i = 1/n$ for example.

2.2. *Randomized low-rank approximation.* Algorithm 2 below computes the leading singular vectors of a smaller matrix S, which is a subsampled and rescaled version of X. Here, the computational savings come from the fact that we only need to compute singular values of a matrix of dimension $m \times s$ with $s \le n$. Recall that computing k leading eigenvectors of a symmetric matrix of dimension s only requires matrix vector products, hence can be performed in $O(ks^2 \log s)$ operations using iterative algorithms such as the

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Algorithm 2 Low-rank approximation

Input: $X \in \mathbf{R}^{m\times n}$ and k, s such that $1 \leq k \leq s < n$.
1: Define a probability vector $q \in \mathbf{R}^n$ such that $q_i = \|X^{(i)}\|_2^2/\|X\|_F^2$, for $i = 1, \ldots, n$.
2: Define a subsampled matrix $S \in \mathbf{R}^{m\times s}$ as follows.
3: **for** $i = 1$ to s **do**
4: Pick an index $j \in [1, n]$ with $\mathbf{P}(j = l) = q_l$.
5: Set $S^{(i)} = X^{(j)}/\sqrt{sq_j}$.
6: **end for**
7: Form the eigenvalue decomposition $S^TS = Y\,\mathbf{diag}(\sigma)Y^T$ where $Y \in \mathbf{R}^{s\times s}$ and $\sigma \in \mathbf{R}^s$.
8: Form a matrix $H \in \mathbf{R}^{m\times k}$ with $H^{(i)} = SY^{(i)}/\sigma_i^{1/2}$.
Output: Approximate singular vectors $H^{(i)}$, $i = 1, \ldots, k$.

power method or Lanczos method. The complexity is in fact $O(kms \log m)$ in our case because an explicit factorization of the matrix is known, (see the appendix for details, as usual we omit the precision target in linear algebra operations, implicitly assuming that it is much finer than ϵ), so that the cost of computing k leading singular vectors of a matrix of size $m \times s$ is $O(ksm \log m)$.

This means that, given the probabilities q_i, the total cost of obtaining k approximate singular vectors using algorithm 2 is $O(ksm \log m)$ instead of $O(knm \log m)$ for exact singular vectors. Of course, computing q_i requires mn operations, but can be done very efficiently in a single pass over the data. We now recall the following result from Drineas, Kannan and Mahoney (2006) which controls the precision of the approximations in algorithm 2.

LEMMA 2. *Let $X \in \mathbf{R}^{m\times n}$ and $1 \leq k \leq s < n$. Given a precision target $\epsilon > 0$, if $s \geq 4/\epsilon^2$ and $H \in \mathbf{R}^{m\times k}$ is computed as in algorithm 2, we have*

$$\mathbf{E}[\|X - H_kH_k^TX\|_2^2] \leq \|X - X_k\|_2^2 + \epsilon\|X\|_F^2$$

and if in addition $s > 4\eta^2/\epsilon^2$ where $\eta = 1 + \sqrt{8\log(1/\beta)}$ for $\beta \in [0, 1]$, then

$$\|X - H_kH_k^TX\|_2^2 \leq \|X - X_k\|_2^2 + \epsilon\|X\|_F^2$$

with probability at least $1 - \beta$, where X_k is the best rank k approximation of X.

PROOF. See Theorem 4 in Drineas, Kannan and Mahoney (2006). □

An identical precision bound holds in the Frobenius norm when $s \geq 4k/\epsilon^2$. We now adapt these results to our setting in the following lemma, which shows how to approximate the spectral radius of a symmetric matrix X using algorithm 2.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

LEMMA 3. *Let $X \in \mathbf{R}^{m\times n}$ and $\beta \in [0,1]$. Given a precision target $\epsilon > 0$, construct a matrix $S \in \mathbf{R}^{m\times s}$ by subsampling the columns of X as in algorithm 2. Let $\eta = 1 + \sqrt{8\log(1/\beta)}$ and*

$$s = \eta^2 \frac{\|X\|_2^2}{\epsilon^2} \mathbf{NumRank}(X)^2 \tag{2}$$

we have

$$\mathbf{E}[|\|S\|_2 - \|X\|_2|] \leq \epsilon$$

and

$$|\|S\|_2 - \|X\|_2| \leq \epsilon$$

with probability at least $1-\beta$.

PROOF. Using the Hoffman-Wielandt inequality (see (Stewart and Sun, 1990, Th. 3.1) or the proof of (Drineas, Kannan and Mahoney, 2006, Th.2) for example) we get

$$|\|S\|_2^2 - \|X\|_2^2| \leq \|SS^T - XX^T\|_F$$

hence

$$|\|S\|_2 - \|X\|_2| \leq \|SS^T - XX^T\|_F / \|X\|_2$$

and Jensen's inequality together with the matrix multiplication result in Lemma 1 yields

$$\mathbf{E}[\|SS^T - XX^T\|_F] \leq \frac{\|X\|_F^2}{\sqrt{s}}$$

and

$$\|SS^T - XX^T\|_F \leq \frac{\eta\|X\|_F^2}{\sqrt{s}}$$

with probability at least $1-\beta$. Combining these two inequalities with the sampling rate in (2)

$$s = \eta^2 \frac{\|X\|_F^4}{\epsilon^2 \|X\|_2^2}$$

yields the desired result. □

The subsampling rate required to achieve a precision target ϵ has a natural interpretation. Indeed

$$s = \eta^2 \frac{\|X\|_2^2}{\epsilon^2} \mathbf{NumRank}(X)^2$$

is simply the squared ratio of the numerical rank of the matrix X over the relative precision target $\epsilon/\|X\|_2$, times a factor η^2 controlling the confidence level. The numerical rank $\mathbf{NumRank}(X)$ always satisfies $1 \leq \mathbf{NumRank}(X) = \|X\|_F^2/\|X\|_2^2 \leq \mathbf{Rank}(X)$ and can be seen as a stable relaxation of the rank of the matrix X (see Rudelson and Vershynin (2007) for a discussion). Note also that, by construction, the subsampled matrix always has lower rank than the matrix X. The expectation bound is still valid if we drop the factor η in (2).

3. Stochastic approximation algorithm. Below, we will use a stochastic approximation algorithm to solve problem (1) when the gradient is approximated using the subsampling algorithms detailed above. We focus on a stochastic approximation of problem (1) written

$$\min_{y\in Q} f(y) \equiv \mathbf{E}\left[\left\|\pi^{(s)}\left(\sum_{j=1}^{p} y_j A_j + C\right)\right\|_2\right] - b^T y \tag{3}$$

in the variable $y \in \mathbf{R}^p$ and parameters $A_j \in \mathbf{S}_n$, for $j = 1,\ldots,p$, $b \in \mathbf{R}^p$ and $C \in \mathbf{S}_n$, with $1 \leq s \leq n$ controlling the sampling rate, where the function $\|\pi^{(s)}(\sum_{j=1}^p y_j A_j + C)\|_2$ and a subgradient with respect to y are computed using algorithms 1 and 2. For $X \in \mathbf{S}_n$, we have written $\pi^{(s)}(X)$ the subsampling/scaling operation used in algorithms 1 and 2 with

$$\pi^{(s)}(X) = S, \tag{4}$$

where $0 < s < n$ controls the sampling rate and $S \in \mathbf{R}^{n\times s}$ is the random matrix defined in algorithm 2 whose columns are a scaled sample of the columns of X. We will write $S = \pi_{(s)}(X)$ the matrix obtained by subsampling rows as in algorithm 1. We also define $\mathcal{A} \in \mathbf{R}^{n^2\times p}$ as the matrix whose columns are given by $\mathcal{A}^{(j)} = \mathbf{vec}(A_j)$, $j = 1,\ldots,p$.

3.1. *Stochastic approximation algorithm.* We show the following lemma approximating the gradient of the function $\|\pi^{(s)}(\sum_{j=1}^p y_j A_j + C)\|_2$ with respect to y and bounding its quadratic variation.

LEMMA 4. *Given $A_j \in \mathbf{S}_n$ with $\mathcal{A} \in \mathbf{R}^{n^2\times p}$ defined as above, for $j = 1,\ldots,p$, $b \in \mathbf{R}^p$, $C \in \mathbf{S}_n$ and sampling rates s_1 and s_2, a (stochastic) subgradient of the function $\|\pi^{(s_1)}(\sum_{j=1}^p y_j A_j + C)\|_2 - b^T y$ with respect to y is given by the vector $w \in \mathbf{R}^p$ with*

$$w = \mathcal{A}^T \mathbf{vec}(vv^T)u - b$$

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

where $v \in \mathbf{R}^n$ *is a leading singular vector of the subsampled matrix* $S = \pi^{(s_1)}(\sum_{j=1}^p y_j A_j + C)$ *formed in algorithm 2 and* $u \in \{-1, 1\}$ *is the sign of the associated eigenvalue. Furthermore, the product* $\mathcal{A}^T \mathbf{vec}(vv^T)$ *can be approximated using algorithm 1 to form an approximate gradient*

$$g = \pi^{(s_2)}(\mathcal{A}^T)\ \pi_{(s_2)}(\mathbf{vec}(vv^T))u - b,$$

which satisfies

(5)

$$\mathbf{E}[g] = \mathcal{A}^T \mathbf{vec}(vv^T)u - b \in \partial f(y) \quad \textit{and} \quad \mathbf{E}[\|g\|_2^2] \leq M_*^2 \equiv 2\frac{\|\mathcal{A}\|_F^2}{s_2} + 2\|b\|_2^2.$$

Proof. Iterated expectations give $\mathbf{E}[g] = \mathbf{E}[w] \in \partial f(y)$. The sampling probabilities q_i used in approximating the matrix vector product $\mathcal{A}^T \mathbf{vec}(vv^T)$ following algorithm 1 are defined as

$$q_i = \frac{\|\mathcal{A}_{(i)}\|_2 |\mathbf{vec}(vv^T)_i|}{\sum_{j=1}^{n^2} \|\mathcal{A}_{(j)}\|_2 |\mathbf{vec}(vv^T)_j|}, \quad i = 1, \ldots, n^2.$$

As in (Drineas, Kannan and Mahoney, 2007, Lemma 3), the quadratic variation of the approximate product $\pi^{(s_2)}(\mathcal{A}^T)\ \pi_{(s_2)}(\mathbf{vec}(vv^T))$ is then given by

$$\mathbf{E}[\|\pi^{(s_2)}(\mathcal{A}^T)\ \pi_{(s_2)}(\mathbf{vec}(vv^T))\|_F^2] = \sum_{i=1}^{n^2} \frac{\|\mathcal{A}_{(i)}\|_2^2 \mathbf{vec}(vv^T)_i^2}{s_2 q_i}.$$

With q_i defined as above, we get

$$\begin{aligned} \sum_{i=1}^{n^2} \frac{\|\mathcal{A}_{(i)}\|_2^2 \mathbf{vec}(vv^T)_i^2}{s_2 q_i} &\leq \frac{\left(\sum_{i=1}^{n^2} \|\mathcal{A}_{(i)}\|_2 |\mathbf{vec}(vv^T)_i|\right)^2}{s_2} \\ &\leq \frac{\|\mathcal{A}\|_F^2 \|vv^T\|_F^2}{s_2} \end{aligned}$$

by the Cauchy-Schwarz inequality, because $\|\mathbf{vec}(vv^T)\|_2^2 = \|vv^T\|_F^2 = \|v\|_2^4 = 1$, hence the desired result. □

Note that this procedure is not advantageous when $\|\mathcal{A}^T\|_F \gg \|\mathcal{A}^T\|_2$ and s_2 is small. We now use this result to produce an explicit bound on the complexity of solving problems (3) and (1) by subsampling using a stochastic approximation algorithm. In this section, we let $\|\cdot\|$ be a general norm on $\mathbf{R}^p$, we write $\|\cdot\|_*$ its dual norm and define $\delta_*(p)$ as the smallest number such that

$\|y\|_2 \leq \delta_*(p)\|y\|_*$ for all $y \in \mathbf{R}^p$. Following the notation in (Juditsky et al., 2009, §2.3), we let $\omega(x)$ be a distance generating function, i.e. a function such that

$$Q^o = \left\{ x \in Q : \ \exists y \in \mathbf{R}^p, \ x \in \operatorname*{argmin}_{u \in Q} [y^T u + \omega(u)] \right\}$$

is a convex set. We assume that $\omega(x)$ is strongly convex on Q^o with modulus α with respect to the norm $\|\cdot\|$, which means

$$(y-x)^T(\nabla\omega(y) - \nabla\omega(x)) \geq \alpha\|y-x\|^2, \quad x, y \in Q^o.$$

We then define a prox-function $V(x,y)$ on $Q^o \times Q$ as follows:

$$V(x,y) \equiv \omega(y) - [\omega(x) + \nabla\omega(x)^T(y-x)],$$

which is nonnegative and strongly convex with modulus α with respect to the norm $\|\cdot\|$. The prox-mapping associated to V is then defined as

$$P_x^{Q,\omega}(y) \equiv \operatorname*{argmin}_{z \in Q} \{ y^T(z-x) + V(x,z) \}. \tag{6}$$

Finally, we define the ω diameter of the set Q as:

$$D_{\omega,Q} \equiv (\max_{z \in Q} \omega(z) - \min_{z \in Q} \omega(z))^{1/2} \tag{7}$$

and we let γ_l for $l = 0, \ldots, N$ be a step size strategy.

The following results control the convergence of the robust stochastic approximation algorithm 3 (see Juditsky et al. (2009), Nesterov (2009), Polyak and Juditsky (1992) or Nemirovsky and Yudin (1983) for further details). We call $\bar{y}$ the optimal solution of problem (3), the lemma below characterizes convergence speed in expectation.

Algorithm 3 Spectral norm minimization using subsampling

Input: Matrices $A_j \in \mathbf{S}_n$, for $j = 1, \ldots, p$, $b \in \mathbf{R}^p$ and $C \in \mathbf{S}_n$, sampling rates s_1 and s_2.

1: Pick initial $y_0 \in Q$
2: **for** $l = 1$ to N **do**
3: Compute $v \in \mathbf{R}^n$, the leading singular vector of the matrix $\pi^{(s_1)}(\sum_{j=1}^p y_{l,j} A_j + C)$, subsampled according to algorithm 2 with $k = 1$ and $s = s_1$.
4: Compute the approximate subgradient $g_l = \pi^{(s_2)}(\mathcal{A}^T)\, \pi_{(s_2)}(\mathbf{vec}(vv^T)) - b$, by subsampling the matrix product using algorithm 1 and $s = s_2$.
5: Set $y_{l+1} = P_{y_l}^{Q,\omega}(\gamma_l g_l)$.
6: Update the running average $\tilde{y}_N = \sum_{k=0}^N \gamma_l y_l / \sum_{k=0}^N \gamma_l$.
7: **end for**

Output: An approximate solution $\tilde{y}_N \in \mathbf{R}^p$ of problem (3) with high probability.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

LEMMA 5. *Given $N > 0$, let M_* be defined as in (5) by*

$$M_*^2 = 2\frac{\|\mathcal{A}\|_F^2}{s_2} + 2\|b\|_2^2,$$

using a fixed step size strategy with

$$\gamma_l = \frac{D_{\omega,Q}}{\delta_*(p)M_*}\sqrt{\frac{2}{\alpha N}}, \quad l = 1, \ldots, N$$

we have, after N iterations of algorithm 3

$$\mathbf{E}[f(\tilde{y}_N) - f(\bar{y})] \leq D_{\omega,Q}\delta_*(p)M_*\sqrt{\frac{2}{\alpha N}}$$

and

$$f(\tilde{y}_N) - f(\bar{y}) \geq \epsilon$$

with probability less than $\frac{D_{\omega,Q}\delta_*(p)M_*}{\epsilon}\sqrt{\frac{2}{\alpha N}}$.

PROOF. By construction $\mathbf{E}[\|g\|_*^2] \leq \delta_*^2(p)M_*^2$, the rest follows from (Juditsky et al., 2009, §2.3) for example. □

Lemma 5 means that we need at most

$$N = \frac{2D_{\omega,Q}^2\delta_*^2(p)M_*^2}{\alpha\epsilon^2\beta^2}$$

iterations to get an ϵ solution to problem (3) with confidence at least $1-\beta$. Typically, the prox function ω and the norm are chosen according to the geometry of Q, to minimize N. The choice of norm also affects $\delta_*(p)$ and obtaining better bounds on M_* in (5) for generic norms would further tighten this complexity estimate.

We now call y^* the solution to the original (deterministic) spectral norm minimization problem (1) and bound the suboptimality of $\tilde{y}_N$ in the (true) problem (1) with high probability.

THEOREM 1. *If the sampling rate s_1 is set to*

$$s_1 = \eta^2\frac{\left\|\sum_{j=1}^p y_j^*A_j + C\right\|_2^2}{\epsilon^2}\mathbf{NumRank}\left(\sum_{j=1}^p y_j^*A_j + C\right)^2 \tag{8}$$

then after

$$N = \frac{2D_{\omega,Q}^2 \delta_*^2(p) M_*^2}{\alpha \epsilon^2 \beta^2} \tag{9}$$

iterations of algorithm 3, we have

$$\left\| \sum_{j=1}^p \tilde{y}_{N,j} A_j + C \right\|_2 - b^T \tilde{y}_N - \left\| \sum_{j=1}^p y_j^* A_j + C \right\|_2 + b^T y^* \leq 2\epsilon$$

with probability at least $1 - \beta$.

PROOF. Recall that we have written y^* the solution to the original (deterministic) problem (1), $\bar{y}$ the solution to the approximate (stochastic) problem (3) and $\tilde{y}_N$ the N-th iterate of algorithm 3 above. Lemma 5 on the convergence of $\tilde{y}_N$ to the solution of the stochastic problem in (3) means

$$f(\tilde{y}_N) - f(\bar{y}) \leq \epsilon$$

with probability at least $1 - \beta$. By definition, $\bar{y}$ minimizes the stochastic problem, so in particular $f(\bar{y}) \leq f(y^*)$, with f the objective value of the *stochastic* problem, so we have in fact

$$f(\tilde{y}_N) - f(y^*) \leq \epsilon. \tag{10}$$

with probability at least $1 - \beta$. Now, with s_1 defined as above, Lemma 3 on the quality of the subsampling approximation to $\|.\|_2$ shows that if the sampling rate is set as in (8) then

$$\mathbf{E}\left[\left| \left\| \sum_{j=1}^p y_j^* A_j + C \right\|_2 - \left\| \pi^{(s)} \left(\sum_{j=1}^p y_j^* A_j + C \right) \right\|_2 \right| \right] \leq \epsilon$$

and Jensen's inequality yields

$$\left| \left\| \left(\sum_{j=1}^p y_j^* A_j + C \right) \right\|_2 - b^T y^* - f(y^*) \right| \leq \epsilon.$$

which bounds the difference between the minimum of the (true) problem in (1) and the value $f(y^*)$ of its stochastic approximation in (3), combining this with inequality (10) we finally get that

$$f(\tilde{y}_N) - \left\| \left(\sum_{j=1}^p y_j^* A_j + C \right) \right\|_2 + b^T y^* \leq 2\epsilon.$$

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

with probability at least $1-\beta$. Applying Jensen's inequality to $\|\cdot\|_2$, using the fact that the subsampling procedure is unbiased, i.e. $\mathbf{E}[\pi^{(s)}(X)] = X$ for any $X \in \mathbf{S}_n$, shows that

$$\begin{aligned}\left\|\sum_{j=1}^p \tilde{y}_{N,j}A_j + C\right\|_2 - b^T\tilde{y}_N &= \left\|\mathbf{E}\left[\pi^{(s)}\left(\sum_{j=1}^p \tilde{y}_{N,j}A_j + C\right)\right]\right\|_2 - b^T\tilde{y}_{N,j}\\ &\leq \mathbf{E}\left[\left\|\pi^{(s)}\left(\sum_{j=1}^p \tilde{y}_{N,j}A_j + C\right)\right\|_2\right] - b^T\tilde{y}_{N,j}\\ &= f(\tilde{y}_N)\end{aligned}$$

where the expectation is taken w.r.t. the sampling probability. Hence

$$f(\tilde{y}_N) - \left\|\left(\sum_{j=1}^p y_j^* A_j + C\right)\right\|_2 + b^T y^* \leq 2\epsilon.$$

implies

$$\left\|\sum_{j=1}^p \tilde{y}_{N,j}A_j + C\right\|_2 - b^T\tilde{y}_N - \left\|\sum_{j=1}^p y_j^* A_j + C\right\|_2 + b^T y^* \leq 2\epsilon$$

which is the desired result. □

This result allows us to bound the *oracle* complexity of solving (1) by subsampling. In practice of course, both the spectral norm and the numerical rank of the solution matrix $\sum_{j=1}^p y_j^* A_j + C$ are unknown and guarantees of successful termination (with high probability) depend on the type of *stopping criterion* available. Given an exact stopping criterion certifying that $y \in \mathbf{R}^p$ is optimal (e.g. a target objective value), we can *search* for the minimum sampling rate in (8) by e.g. starting from a low target and doubling the sampling rate until we obtain an optimal solution. On the other hand, if an exact stopping criterion is not available and a more conservative stopping condition is used (e.g. the surrogate duality gap detailed in §3.3) it is possible for the algorithm to become more expensive than standard subgradient techniques. These two scenarios are detailed below.

- **Exact stopping criterion.** We assume here that we have a reasonably efficient test for the optimality of the current iterate $\tilde{y}_N$, e.g. a specific target for the objective value below which the optimization procedure can be stopped. In this case, even if we have no a priori

knowledge of the rank of the solution matrix, we can search for it, starting with a low guess. The simple lemma below explicitly summarizes the complexity of this procedure.

LEMMA 6. *Suppose we start from a sampling rate $s = 1$ and run algorithm 3 repeatedly, doubling the sampling rate until the stopping criterion certifies the solution is optimal. Then, with probability at least $1 - \lceil \log_2(s_1) \rceil \beta$, algorithm 3 needs to be run at most*

$$\lceil \log_2(s_1) \rceil$$

times, where s_1 is given in (8), before finding an optimal solution to (1).

PROOF. Starting from $s = 1$, we simply need to double the sampling rate at most $\lceil \log_2(s_1) \rceil$ before it becomes larger than s_1. At the sampling rate $s = s_1$, algorithm 3 will produce an optimal solution with prob. $1 - \beta$. □

In this scenario, Lemma 5 shows that the number of iterations required to reach a target precision ϵ with confidence greater than $1 - \beta \lceil \log_2(s_1) \rceil$ grows as

$$N^{total} = O\left(\frac{\lceil \log_2(s_1) \rceil D^2_{\omega,Q} \delta^*(p)^2 \left(\|\mathcal{A}\|_F^2 / s_2 + \|b\|_2^2 \right)}{\alpha \epsilon^2 \beta^2} \right) \tag{11}$$

where s_1 is given in (8), and the cost of each iteration is detailed in §3.2 below and the overall complexity of the method is summarized in Table 1. In fact, we will see in §3.2 that the complexity of each iteration is dominated by a term $O(sn \log n)$, where s is the sampling rate, and because

$$\sum_{i=1}^{\lceil \log_2(s_1) \rceil} 2^i \leq 2^{\lceil \log_2(s_1) \rceil + 1} \leq 4 s_1$$

we then observe that searching for the minimal sampling rate by repeatedly solving (3) for increasing sampling rates will be less than four times as expensive as solving the problem in the oracle case.

- **Conservative stopping criterion.** Typically, producing a conservative stopping oracle means computing a surrogate duality gap and we will show in §3.3 how this can be done efficiently in some examples. In

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

this case however, it is possible for a conservative stopping criterion to repeatedly fail to detect optimality when searching for the sampling rate s_1. The complexity of the subsampling algorithm can then become larger than that of the basic subgradient method.

To summarize, when an exact stopping criterion is available, the complexity of finding an optimal solution is equivalent to that of the oracle complexity described in Theorem 1 and the total number of iterations is bounded by (11). However, when only a conservative stopping condition is available, the algorithm can become more expensive than the classical subgradient method. Note that this early stopping issue is shared by many first-order algorithms, as the theoretical upper bounds available for most first-order methods are usually overly conservative, often by one or two orders of magnitude (Nesterov, 2007, §6). In these cases too, a conservative stopping criterion is often used to stop the algorithm early. The next section provides a detailed analysis of the complexity of an iteration of algorithm 3 as a function of ϵ, s_1 and s_2 and the problem data.

3.2. *Complexity.* We now study in detail the complexity of algorithm 3. Suppose we are given a precision target ϵ and fix the sampling rate s_2 arbitrarily between 1 and n^2, with the sampling rate s_1 set as in Theorem 1. The cost of each iteration in algorithm 3 breaks down as follows.

- On line 3: Computing the leading singular vector v, using algorithm 2 with $k = 1$. This means first forming the matrix $(\sum_{j=1}^{p} y_{l,j} A_j + C)$ and computing the probabilities q_i at a cost of $O(n^2)$ operations. Forming the matrix $S = \pi^{(s_1)}(\sum_{j=1}^{p} y_{l,j} A_j + C)$ costs $O(ns_1)$ operations. It remains to compute the leading singular vector of S using the Lanczos method at a cost of $O(s_1 n \log n)$ (cf. §5.1 for details). The total numerical cost of this step is then bounded by $c_1 n^2 + c_2 n s_1$ where c_1 and c_2 are absolute constants. Here, c_1 is always less than ten while c_2 is the number of iterations required by the Lanczos method to reach a fixed precision target (typically 1e-8 or better here) hence we have $c_1 \ll c_2$.
- On line 4: Computing the approximate subgradient

$$g_l = \pi^{(s_2)}(\mathcal{A}^T)\ \pi_{(s_2)}(\mathbf{vec}(vv^T)) - b,$$

by subsampling the matrix product using algorithm 1. This means again forming the vector q at a cost of $O(n^2)$ (the row norms of $\mathcal{A}$ can be precomputed). Computing the subsampled matrix vector product then costs $O(ps_2)$. Both of these complexity bounds have low constants.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

TABLE 1
Complexity of solving problem (1) using subsampled stochastic approximation method versus original algorithm. Here $c_1, \ldots, c_4$ are absolute constants with $c_1, c_3 \ll c_2, c_4$

Complexity	Stoch. Approx.	Stoch. Approx. with Subsampling
Per Iter.	$c_4 n^2 p + c(p)$	$c_2 n \log n\, \eta^2 \frac{\|Y^*\|_2^2}{\epsilon^2} \mathbf{NumRank}(Y^*)^2$ $+c_1 n^2 + c_3 p s_2 + c(p)$
Num. Iter.	$\frac{2D^2_{\omega,Q}\delta^*(p)^2(\|\mathcal{A}^T\|_2^2+\|b\|_2^2)}{\alpha\epsilon^2\beta^2}$	$\frac{2D^2_{\omega,Q}\delta^*(p)^2\left(\|\mathcal{A}\|_F^2/s_2+\|b\|_2^2\right)}{\alpha\epsilon^2\beta^2}$

- On line 5: Computing the projection $y_{l+1} = P_{y_l}^{Q,\omega}(\gamma_l g_l)$, whose numerical cost will be denoted by $c(p)$.

Let us remark in particular that all $O(n^2)$ operations above only require one pass over the data, which means that the entire data set does not need to fit in memory. Using the bound on the quadratic variation of the gradient computed in Lemma 4, we can then bound the number of iterations required by algorithm 3 to produce a ϵ-solution to problem (1) with probability at least $1-\beta$. Let us call $Y^* = \sum_{j=1}^p y_j^* A_j + C$, and recall that $\eta = 1 + \sqrt{8\log(1/\beta)}$, Table 1 summarizes these complexity bounds and compares them with complexity bounds for a stochastic approximation algorithm without subsampling.

We observe that subsampling affects the complexity of solving problem (1) in two ways. Decreasing the (matrix product) subsampling rate $s_2 \in [1, n^2]$ decreases the cost of each iterations but increases the number of iterations in the same proportion, hence has no explicit effect on the total complexity bound. In practice of course, because of higher cache memory speed and better bandwidth on smaller problems, cheaper iterations tend to run more efficiently than more complex ones. Note that the second subsampling step detailed in Lemma 4 is not advantageous when $\|\mathcal{A}^T\|_F \gg \|\mathcal{A}^T\|_2$. When this second subsampling step is skipped the term $\left(\|\mathcal{A}\|_F^2/s_2 + \|b\|_2^2\right)$ is replaced by $\left(\|\mathcal{A}^{\mathcal{T}}\|_2^2 + \|b\|_2^2\right)$ in the bound on the number of iterations, and the term $c_3 p s_2$ becomes $c_3 p n^2$ (or less if the matrices are structured).

The impact of the (singular vector) subsampling rate $s_1 \in [1, n]$ is much more important however, since computing the leading eigenvector of the current iterate is the most complex step in the algorithm when solving problem (1) using stochastic approximation. Because $c_1, c_3 \ll c_2$, the complexity per iteration of solving large-scale problems essentially follows

$$n \log n\, \eta^2 \frac{\|Y^*\|_2^2}{\epsilon^2} \mathbf{NumRank}(Y^*)^2$$

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

hence explicitly depends on both the numerical rank of the solution matrix $Y^* = \sum_{j=1}^{p} y_j^* A_j + C$ and on the relative precision target $\epsilon/\|Y^*\|_2$. This means that problems with simpler solutions will be solved more efficiently than problems whose solutions has a high rank.

The choice of norm $\|\cdot\|$ and distance generating function also has a direct impact on complexity through $c(p)$ and $\delta_*(p)M_*$. Unfortunately here, subsampling error bounds are only available in the Frobenius and spectral norms hence part of the benefit of choosing optimal norm/distance generating function combinations is sometimes lost in the norm ratio bound $\delta_*(p)$. However, choosing a norm/prox function combination according to the geometry of Q can still improve the complexity bound compared to a purely Euclidean setting.

Finally, subsampling could have a more subtle effect on complexity. By construction, solutions to problem (1) tend to have multiple leading singular values which coalesce near the optimum. Introducing noise by subsampling can potentially break this degeneracy and increase the gap between leading eigenvalues. Since the complexity of the algorithm depends in great part on the complexity of computing a leading singular vector using iterative methods such as the power method or the Lanczos method (cf. Appendix), and the complexity of these methods decreases as the gap between the two leading singular values increases, subsampling can also improve the efficiency of iterative singular value computations. However, outside of simple perturbative regimes, not much is understood at this point about the effect of subsampling on the spectral gap.

3.3. *Surrogate duality gap.* In practice, we often have no a priori knowledge of $\mathbf{NumRank}(Y^*)$[2] and if the sampling rate s is set too low, it's possible for the algorithm to terminate at a suboptimal point Y where the subsampling error is less than ϵ (if the error at the true optimal point Y^* is much larger than ϵ). In order to search for the optimal sampling rate s as in Lemma 6, we first need to check for optimality in (1) and we now show how to track convergence in algorithm 3 by computing a surrogate duality gap, at a cost roughly equivalent to that of computing a subgradient. The dual of problem (1) is written

$$\begin{array}{lll} \text{maximize} & \mathbf{Tr}(CX) - S_Q(w) & \\ \text{subject to} & w_j = b_j - \mathbf{Tr}(A_j X), & j = 1, \ldots, p \\ & \|X\|_{\text{tr}} \leq 1, & \end{array} \tag{12}$$

in the variables $X \in \mathbf{S}_n$ and $w \in \mathbf{R}^p$, where $S_Q(v)$ is the support function of the set Q, defined as

$$S_Q(w) \equiv \max_{y \in Q} w^T y.$$

For instance, when Q is an Euclidean ball of radius B, problem (12) becomes

$$\begin{array}{ll} \text{maximize} & \mathbf{Tr}(CX) - B\|w\|_2 \\ \text{subject to} & w_j = b_j - \mathbf{Tr}(A_j X), \quad j = 1, \ldots, p \\ & \|X\|_{\text{tr}} \leq 1, \end{array} \tag{13}$$

in the variables $X \in \mathbf{S}_n$ and $w \in \mathbf{R}^p$. The leading singular vector v in algorithm 3 always satisfies $\|vv^T\|_{\text{tr}} \leq 1$, hence we can track convergence in solving (1) by computing the following surrogate duality gap

$$\left\| \sum_{j=1}^p y_j A_j + C \right\|_2 - b^T y - v^T C v + S_Q(w) \tag{14}$$

where $w_j = b_j - v^T A_j v$ for $j = 1, \ldots, p$.

3.4. *Minimizing the sum of the k largest singular values.* Motivated by applications in statistical learning, we now discuss direct extensions of the results above to the problem of minimizing the sum of the k largest singular values of an affine combination of matrices, written

$$\min_{y \in Q} \ \sum_{i=1}^k \sigma_i \left(\sum_{j=1}^p y_j A_j + C \right) - b^T y \tag{15}$$

in the variable $y \in \mathbf{R}^p$, with parameters $A_j \in \mathbf{S}_n$, for $j = 1, \ldots, p$, $b \in \mathbf{R}^p$ and $C \in \mathbf{S}_n$. As in the previous section, we also form its stochastic approximation

$$\min_{y \in Q} f(y) \equiv \mathbf{E}\left[\sum_{i=1}^k \sigma_i \left(\pi^{(s)} \left(\sum_{j=1}^p y_j A_j + C \right) \right) \right] - b^T y \tag{16}$$

in the variable $y \in \mathbf{R}^p$, with $1 \leq s \leq n$ controlling the sampling rate. We now prove an analog of Lemma 3 for this new objective function.

LEMMA 7. *Let $X \in \mathbf{R}^{m \times n}$ and $\beta \in [0,1]$. Given a precision target $\epsilon > 0$, $k \geq 1$ and a matrix $S \in \mathbf{R}^{m \times s}$ constructed by subsampling the columns of X as in algorithm 2, let $\eta = 1 + \sqrt{8 \log(1/\beta)}$ and*

$$s = \eta^2 \frac{(\sum_{i=1}^k \sigma_i(X))^2}{\epsilon^2} \frac{\mathbf{NumRank}(X)^2}{k^2} \kappa(X)^4 \, \mathbf{Rank}(X) \tag{17}$$

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

where $\kappa(X) = \sigma_1(X)/\sigma_r(X)$ *with* $r = \min\{k, \mathbf{Rank}(X)\}$, *we have*

$$\mathbf{E}\left[\sum_{i=1}^{k} |\sigma_i(X) - \sigma_i(S)|\right] \leq \epsilon$$

and

$$\sum_{i=1}^{k} |\sigma_i(X) - \sigma_i(S)| \leq \epsilon$$

with probability at least $1 - \beta$.

PROOF. Because $\mathbf{Rank}(SS^T) \leq \mathbf{Rank}(XX^T)$ by construction, we always have

$$\begin{aligned}\sum_{i=1}^{k} \left|\sigma_i^2(X) - \sigma_i^2(S)\right| &= \sum_{i=1}^{k} |\sigma_i(X) - \sigma_i(S)|\,(\sigma_i(X) + \sigma_i(S)) \\ &\geq \sigma_r(X) \sum_{i=1}^{k} |\sigma_i(X) - \sigma_i(S)|\end{aligned}$$

where $r = \min\{k, \mathbf{Rank}(X)\}$. Because the sum of the k largest singular values is a unitarily invariant norm on $\mathbf{S}_n$ (see (Horn and Johnson, 1991, §3.4)), Mirsky's theorem (see (Stewart and Sun, 1990, Th. 4.11) for example) shows that

$$\begin{aligned}\sum_{i=1}^{k} \left|\sigma_i^2(X) - \sigma_i^2(S)\right| &= \sum_{i=1}^{k} \left|\sigma_i(XX^T) - \sigma_i(SS^T)\right| \\ &\leq \sum_{i=1}^{k} \sigma_i(XX^T - SS^T)\end{aligned}$$

and because, by construction, the range of SS^T is included in the range of XX^T, we must have $\mathbf{Rank}(XX^T - SS^T) \leq \mathbf{Rank}(XX^T)$ and

$$\sum_{i=1}^{k} \sigma_i(XX^T - SS^T) \leq \sqrt{\mathbf{Rank}(X)}\ \|XX^T - SS^T\|_F$$

Jensen's inequality together with the matrix multiplication result in Lemma 1 yield

$$\mathbf{E}[\|SS^T - XX^T\|_F] \leq \frac{\|X\|_F^2}{\sqrt{s}}$$

and

$$\|SS^T - XX^T\|_F \leq \frac{\eta \|X\|_F^2}{\sqrt{s}}$$

with probability at least $1-\beta$. Combining these inequalities with the sampling rate in (17)

$$s = \eta^2 \frac{\|X\|_F^4 \, \mathbf{Rank}(X)}{\epsilon^2 \sigma_r(X)^2}$$

and using

$$\frac{\|X\|_F^4}{(\sum_{i=1}^k \sigma_i(X))^2 \sigma_r(X)^2} \leq \frac{\mathbf{NumRank}(X)^2}{k^2} \kappa(X)^4$$

yields the desired result. □

Once again, the subsampling rate in the above lemma has a clear interpretation,

$$s_1 = \eta^2 \frac{(\sum_{i=1}^k \sigma_i(X))^2}{\epsilon^2} \frac{\mathbf{NumRank}(X)^2}{k^2} \kappa(X)^4 \, \mathbf{Rank}(X)$$

is the product of a term representing relative precision, a term reflecting the rank of X and a term in $\kappa(X)$ representing its (pseudo) condition number. Note that the bound can be further refined when $\sigma_r \leq \epsilon$. Lemma 7 allows us to compute the gradient by subsampling when using algorithm 3 to solve problem (15). The remaining steps in the algorithm are identical, except that the matrix vv^T is replaced by a combination of matrices formed using the k leading singular vectors, with Frobenius norm $\sqrt{k}$. The cost of each iteration is dominated by the term

$$c_2 k s_1 m \log n \; + c_1 nm + c_3 p s_2 + c(p)$$

with s_1 defined above, and the total number of iterations growing as

$$\frac{2D_{\omega,Q}^2 \delta^*(p)^2 \left(k\|\mathcal{A}\|_F^2/s_2 + \|b\|_2^2\right)}{\alpha \epsilon^2 \beta^2}.$$

When $\|\mathcal{A}^T\|_F \gg \|\mathcal{A}^T\|_2$ and s_2 is small, the second subsampling step detailed in Lemma 4 is skipped and the cost per iteration becomes

$$c_2 k s_1 m \log n \; + c_1 nm + c_3 pmn + c(p)$$

and the total number of iterations grows as

$$\frac{2D_{\omega,Q}^2 \delta^*(p)^2 \left(k\|\mathcal{A}^T\|_2^2 + \|b\|_2^2\right)}{\alpha \epsilon^2 \beta^2}$$

because $M_*^2 \leq 2\left(k\|\mathcal{A}^T\|_2^2 + \|b\|_2^2\right)$ in this case.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

4. Applications & numerical results. In this section, we first detail a few instances of problem (1) arising in statistical learning. We then study the numerical performance of the methods detailed here on large scale problems.

4.1. *Spectral norm minimization.* For a given matrix $A \in \mathbf{S}_n$, we begin by studying a simple instance of problem (1) written

$$\begin{array}{ll} \text{minimize} & \|A+U\|_2 \\ \text{subject to} & |U_{ij}| \leq \rho, \quad i,j=1,\ldots,n \end{array} \tag{18}$$

in the matrix $U \in \mathbf{S}_n$. This problem is closely related to a relaxation for sparse PCA (see d'Aspremont et al. (2007)) and we use it in the next section to test the numerical performance of algorithm 3. The complexity of the main step in the algorithm (i.e. computing the gradient) is controlled by the sampling rate in Lemma 3, which is written

$$s_1 = \eta^2 \frac{\|A+U^*\|_2^2}{\epsilon^2} \mathbf{NumRank}(A+U^*)^2$$

where $U^* \in \mathbf{S}_n$ is the optimal solution to problem (18). The prox function used here is the square Euclidean norm, and the prox-mapping is then a simple Euclidean projection on the box $[-\rho,\rho]^{n^2}$. The cost of each iteration is then dominated by the term

$$c_2 s_1 n \log n \; + (c_1+c_3+1)n^2.$$

with s_1 defined above, and the total number of iterations grows as

$$\frac{4\lceil \log_2(s_1)\rceil n^2\rho^2}{\epsilon^2\beta^2},$$

because the gradient always has norm one in this problem and the second subsampling step in Lemma 4 is not beneficial.

4.2. *Matrix factorization and collaborative filtering.* Matrix factorization methods have been heavily used to solve collaborative filtering problems (e.g. the *Netflix* problem) and we refer the reader to Srebro (2004), Bach (2007), Recht, Fazel and Parrilo (2007) or Candes and Recht (2008) for details. Srebro (2004) focuses on the following problem instance

$$\text{minimize} \quad \|X\|_{\text{tr}} + c \sum_{(i,j)\in S} \max(0, 1 - X_{ij}M_{ij}) \tag{19}$$

in the variable $X \in \mathbf{R}^{m\times n}$, where M is a sparse matrix of ratings, S is the set of known ratings (typically small), and $c > 0$ is a parameter controlling the rank versus accuracy tradeoff. Here, the trace norm can be understood as a convex lower bound on the rank function (as in Fazel, Hindi and Boyd (2001)) but sometimes also has a direct interpretation in terms of learning (see Srebro (2004)). The dual of this problem is written

$$\begin{array}{ll} \text{maximize} & \sum_{ij} Y_{ij} \\ \text{subject to} & \|Y \circ M\|_2 \leq 1 \\ & 0 \leq Y_{ij} \leq c \end{array}$$

in the variable $Y \in \mathbf{R}^{n\times m}$, where $Y \circ M$ is the Schur (componentwise) product of Y and M. Because M is usually sparse, this problem is typically sparse too (i.e. most of the coefficients of Y can be set to c). This last problem is equivalent to

$$\begin{array}{ll} \text{minimize} & \|Y \circ M\|_2 \\ \text{subject to} & \sum_{ij} Y_{ij} = 1 \\ & 0 \leq Y_{ij} \leq d \end{array}$$

for some $d > 0$, which is a particular instance of (1). In the numerical experiments that follow, we focus on a simpler formulation of (15) written

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^k \sigma_i(X) \\ \text{subject to} & X_{ij} = M_{ij}, \quad (i,j) \in S \\ & |X_{ij}| \leq B \end{array} \tag{20}$$

in the variable $X \in \mathbf{R}^{m\times n}$, for some $k \in [1, n]$ and $B > 0$, which is also a particular instance of problem (15). We assume $n \geq m$. This replaces the hinge-loss penalty in problem (19) with equality constraints. The classic trace heuristic uses $\|X\|_{\text{tr}}$ instead of $\sum_{i=1}^k \sigma_i(X)$ but we will see in what follows that minimizing the later term also tends to produce low rank solutions.

In this particular case, the complexity of the main step in the algorithm (i.e. computing the gradient) is controlled by the sampling rate in Lemma 7, which can be simplified here to

$$s_1 = \eta^2 \frac{\|Y^*\|_{\text{tr}}^2}{\epsilon^2} \kappa(Y^*)^2 \, \mathbf{Rank}(Y^*)$$

where $Y^* = \sum_{j=1}^p y_j^* A_j + C$ and $\kappa(Y^*) = \sigma_1(Y^*)/\sigma_r(Y^*)$ with $r = \mathbf{Rank}(Y^*)$. The prox function used here is the square Euclidean norm, and the prox-mapping is then a simple Euclidean projection on the box $[-B, B]^{m\times n}$ for

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

the coefficients whose rating is not given. As in (3.4), the cost of each iteration is dominated by the term

$$c_2 s_1 m \log n + (c_1 + c_3 + 1)nm.$$

with s_1 defined above, and the total number of iterations growing as

$$\frac{4\lceil \log_2(s_1)\rceil nmB^2k}{\epsilon^2\beta^2},$$

because the gradient has Frobenius norm $\sqrt{k}$ in this case. This bound can be further refined when $\sigma_r \leq \epsilon$. In practice, the complexity of solving problem (20) can often be further reduced using the simple observation that an optimal solution of (15) will also be optimal in (20) whenever $\mathbf{Rank}(Y_k^*) < k$, where Y_k^* is the optimal solution to (15) here. Once again, the sampling rate s has a natural interpretation as the product of a relative precision term, a term reflecting the condition number of the solution and the rank of the optimal solution. It means in particular that problems whose solutions have a lower rank are explicitly easier to solve than problems with more complex (higher rank) solutions. Of course, much faster specialized algorithms are available for this problem, but those methods that exploit the fact that the solution is low-rank (like alternating minimization) are non-convex.

4.3. *Numerical experiments.* In this section, we test the quality of the subsampling approximations detailed in Section 2 on various matrices. We also evaluate the performance of the algorithms detailed above on large scale problem instances. Numerical code reproducing these experiments is available from the author's webpage.

Randomized low-rank approximations. Here, we first measure the quality of the randomized low-rank matrix approximation on both randomly generated matrices and on covariance matrices formed using gene expression data. Because the spectrum of naive large scale random matrices is very structured, these examples are too simple to appropriately benchmark numerical error in algorithm 2. Fortunately, as we will see below, generating random symmetric matrices with a given spectral measure is straightforward.

Suppose $X \in \mathbf{S}_n$ is a matrix with normally distributed coefficients, $X_{ij} \sim \mathcal{N}(0,1)$, $i,j = 1,\ldots,n$. If we write its QR decomposition, $X = QR$ with $Q,\ R \in \mathbf{R}^{n\times n}$, then the orthogonal matrix Q is Haar distributed on the orthogonal group $\mathcal{O}_n$ (see Diaconis (2003) for example). This means that to generate a random matrix with given spectrum $\mu \in \mathbf{R}^n$, we generate a normally distributed matrix X, compute its QR decomposition and the matrix

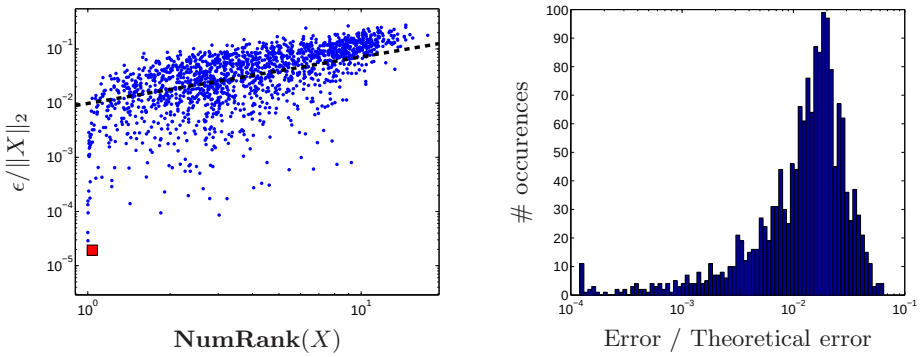


FIG 1. *Left: Loglog plot of relative error* $\epsilon/\|X\|_2$ *versus numerical rank* $\mathbf{NumRank}(X)$ *with 20% subsampling and* $n = 500$ *on random matrices (blue dots) and gene expression covariance (red square). The dashed line has slope one in loglog scale. Right: Histogram plot in semilog scale of relative error* $\epsilon/\|X\|_2$ *over theoretical bound* $\eta\,\mathbf{NumRank}(X)/\sqrt{s}$ *for random matrices with* $n = 500$.

$Q\,\mathbf{diag}(\mu)Q^T$ will be uniformly distributed on the set of symmetric matrices with spectrum μ. Because the spectral measure of "natural" covariance matrices often follows a power law (Tracy-Widom in the Gaussian case, see Johnstone (2001) and El Karoui (2007) for a discussion), we sample the spectrum μ from a beta distribution with various exponents to get realistic random matrices with a broad range of numerical ranks. We also use a covariance matrix formed using the gene expression data set in Alon et al. (1999).

In Figure 1, we plot relative error $\epsilon/\|X\|_2$ against the numerical rank $\mathbf{NumRank}(X)$ in loglog scale with 20% subsampling and $n = 500$ on random matrices generated as above and on the gene expression covariance from Alon et al. (1999). We notice that, on these experiments, the relative error grows at most linearly with the numerical rank of the matrix, as predicted by Lemma 3. We then plot the histogram in semilog scale of relative error $\epsilon/\|X\|_2$ over theoretical bound $\eta\,\mathbf{NumRank}(X)/\sqrt{s}$ for random matrices with $n = 500$. In Figure 2, we plot relative error $\epsilon/\|X\|_2$ versus sampling rate s, in loglog scale, for a gene expression covariance with $n = 500$. Once gain, the error decreases as $1/\sqrt{s}$ as predicted by Lemma 3. We also plot the median speedup factor (over ten runs) in computing largest magnitude eigenvalues using ARPACK with and without subsampling on a gene expression covariance matrix with $n = 2000$, for various values of the sampling ratio s/n. Note that both exact and subsampled eigenvalues are computed using direct MEX calls to ARPACK by Lehoucq, Sorensen and Yang (1998), as `eigs` (MATLAB's interface to ARPACK) carries a massive overhead. In all the experiments above, the confidence level used in computing η was set to 99%.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

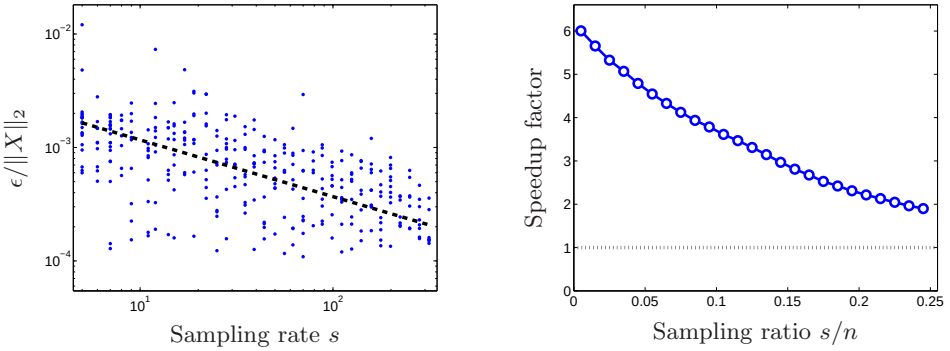


FIG 2. *Left: Loglog plot of relative error $\epsilon/\|X\|_2$ versus sampling rate s for a gene expression covariance with $n = 500$. The dashed line has slope -1/2 in loglog scale. Right: Plot of median speedup factor in computing largest magnitude eigenvalue, using ARPACK with and without subsampling on a gene expression covariance matrix with $n = 2000$, for various values of the sampling ratio s/n.*

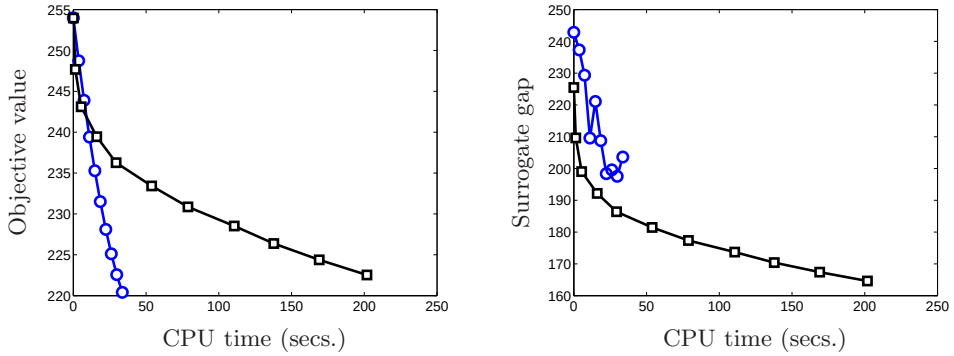


FIG 3. *Left: Objective value versus CPU for a sample matrix factorization problem in dimension 100, using a deterministic gradient (squares) or a subsampled gradient with subsampling rate set at 20% (circles). Right: Surrogate duality gap versus CPU time on the same example.*

Stochastic approximation with subsampling. In Figure 3, we generate a sample ratings matrix $X = VV^T$ for the collaborative filtering problem (20) in §4.2, where V is a discrete feature matrix $V \in [0, 4]^{n\times 3}$, with $n = 100$. We "observe" only 30% of the coefficients in V and solve problem (20) with $k = 4$ and $B = 10$ to approximately reconstruct the full ratings matrix. We plot objective value versus CPU time in seconds for this sample matrix factorization problem, using a stochastic approximation algorithm with deterministic gradient or the subsampled gradient algorithm 3 with subsampling ratio s_1/n set at 20%. We also plot surrogate duality gap versus CPU

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

TABLE 2
CPU time (in seconds) versus problem dimension n for deterministic and subsampled stochastic approximation algorithms on spectral norm minimization problems

n	Deterministic	Subsampling	Speedup factor
500	5	5	0.92
750	19	13	1.40
1000	32	24	1.31
1500	107	58	1.84
2000	281	120	2.34

TABLE 3
Median CPU time (in seconds) versus problem dimension n for deterministic and subsampled stochastic approximation algorithms on collaborative filtering problems

n	Deterministic	Subsampling	Speedup factor
100	154	23	6.67
200	766	63	12.2
500	4290	338	12.7

time on the same example. We notice that while the subsampled algorithm converges much faster than the deterministic one, the quality of the surrogate dual points and duality gap produced using subsampled gradients as in §3.3 is worst than in the deterministic case.

In Table 2, using the same 20% sampling rate we compare CPU time versus problem dimension n for subsampled and deterministic algorithms when solving the following instance of problem (1)

$$\begin{array}{ll} \text{minimize} & \|C+X\|_2 \\ \text{subject to} & \|X\|_\infty \leq \rho \end{array}$$

in the variable $X \in \mathbf{S}_n$ where C is a covariance matrix constructed using the n variables in the gene expression data set of Alon et al. (1999) with maximum variance, for various values of n, and $\rho = 1/2$. We run 200 iterations of the deterministic algorithm and run the stochastic algorithm until it reaches the best value found by the deterministic method. Finally, using random ratings data generated as above, we solve sample collaborative filtering solve problems (20) with $k = 4$ and $B = 10$ for ratings matrix of various dimensions n. We run 5000 iterations of the deterministic algorithm and run the stochastic algorithm until it reaches the best value found by the deterministic method.

Of course, specialized software packages solve much larger problems and this example is only here to illustrate the potential speedup. We report median CPU time over ten sample problems in Table 3. Here, subsampling speeds up the algorithm by an order of magnitude, however the stochastic

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

approximation algorithm is still not competitive with (non convex) local minimization techniques over low rank matrices.

5. Appendix. The complexity results detailed above heavily rely on the fact that extracting *one* leading eigenvector of a symmetric matrix $X \in \mathbf{S}_n$ can be done by computing a few matrix vector products. While this simple fact is easy to prove using the power method when the eigenvalues of X are well separated, the problem becomes significantly more delicate when the spectrum of X is clustered. The section that follows briefly summarizes how modern numerical methods solve this problem in practice.

5.1. *Computing one leading eigenvector of a symmetric matrix.* We start by recalling how packages such as LAPACK Anderson et al. (1999) form a full eigenvalue (or Schur) decomposition of a symmetric matrix $X \in \mathbf{S}_n$. The algorithm is strikingly stable and, despite its $O(n^3)$ complexity, often competitive with more advanced techniques when the matrix X is small. We then discuss the problem of approximating one leading eigenpair of X using Krylov subspace methods with complexity growing as $O(n^2 \log n)$ with the dimension (or less when the matrix is structured).

Full eigenvalue decomposition. Full eigenvalue decompositions are computed by first reducing the matrix X to symmetric tridiagonal form using Householder transformations, then diagonalizing the tridiagonal factor using iterative techniques such as the QR or divide and conquer methods for example (see (Stewart, 2001, Chap. 3) for an overview). The classical QR algorithm (see (Golub and Van Loan, 1990, §8.3)) implicitly relied on power iterations to compute the eigenvalues and eigenvectors of a symmetric tridiagonal matrix with complexity $O(n^3)$, however more recent methods such as the MRRR algorithm by Dhillon and Parlett (2003) solve this problem with complexity $O(n^2)$. Starting with the third version of LAPACK, the MRRR method is part of the default routine for diagonalizing a symmetric matrix and is implemented in the `STEGR` driver (see Dhillon, Parlett and Vömel (2006)).

Overall, the complexity of forming a *full* Schur decomposition of a symmetric matrix $X \in \mathbf{S}_n$ is then $4n^3/3$ flops for the Householder tridiagonalization, followed by $O(n^2)$ flops for the Schur decomposition of the tridiagonal matrix using the MRRR algorithm.

Computing one leading eigenpair. We now give a brief overview of the complexity of computing leading eigenpairs using Krylov subspace methods and we refer the reader to (Stewart, 2001, §4.3), (Golub and Van Loan, 1990, §8.3, §9.1.1) or Saad (1992) for a more complete discussion. Let $u \in \mathbf{R}^n$ be

a given vector, we form the following *Krylov* sequence

$$\left\{u, Xu, X^2u, \ldots, X^ku\right\}$$

by computing k matrix vector products. If we call $\mathcal{K}_k(X,u)$ the subspace generated by these vectors and write $X = \sum_{i=1}^n \lambda_i x_i x_i^T$ a spectral decomposition of X, assuming, for now, that

$$\lambda_1 > \lambda_2 \geq \cdots \geq \lambda_n,$$

one can show using Chebyshev polynomials (see e.g. (Stewart, 2001, §4.3.2) for details) that

$$\tan\angle\left(x_1, \mathcal{K}_k(X,u)\right) \lesssim \frac{\tan\angle(x_1,u)}{\left(1+2\sqrt{\eta+\eta^2}\right)^{k-1}} \quad \text{where } \eta = \frac{\lambda_1-\lambda_2}{\lambda_2-\lambda_n},$$

in other words, after a few iterations, Krylov subspaces contain excellent approximations of leading eigenpairs of X.

This result is exploited by the Lanczos procedure to extract approximate eigenpairs of X called *Ritz* pairs (see (Golub and Van Loan, 1990, Chap. 9) or (Stewart, 2001, §5.1.2) for a complete discussion). In practice, the matrix formed by the Krylov sequence is very ill-conditioned (as X^ku gets increasingly close to the leading eigenvector) so the Lanczos algorithm simultaneously updates an orthogonormal basis for $\mathcal{K}_k(X,u)$ and a *partial* tridiagonalization of X. The Lanczos procedure is described in Algorithm 4 and requires k matrix vector products and an additional $4nk$ flops. Note that the *only* way in which the data in X is accessed is through the matrix vector products Xu_j.

Algorithm 4 Lanczos decomposition

Input: Matrices $X \in \mathbf{S}_n$ and initial vector $u_1 \in \mathbf{R}^n$.

1: Set $u_0 = 0$ and $\beta_0 = 0$.
2: **for** $j = 1$ to k **do**
3: Compute $v = Xu_j$.
4: Set $\alpha_j = u_j^T v$.
5: Update $v = v - \alpha_j u_j - \beta_{j-1}u_{j-1}$.
6: Set $\beta_j = \|v\|_2$.
7: Set $u_{j+1} = v/\beta_j$.
8: **end for**

Output: A Lanczos decomposition

$$XU_k = U_kT_k + \beta_k u_{k+1}e_k^T,$$

where $U_k \in \mathbf{R}^{n\times k}$ is orthogonal and $T_k \in \mathbf{S}_k$ is symmetric tridiagonal.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

In theory, one could then diagonalize the matrix T_k (which costs $O(k^2)$ using the MRRR algorithm as we have seen above) to produce Ritz vectors. In practice, key numerical difficulties often arise. First, finite precision arithmetics cause a significant loss of orthogonality in U_k. This is remedied by various reorthogonalization strategies (cf. (Stewart, 2001, §5.3.1)). A more serious problem is clustered or multiple eigenvalues in the spectrum periphery. In fact, it is easy to see that Krylov subspace methods cannot isolate multiple eigenvalues. Assume that the leading eigenvalue has multiplicity two for example, we then have

$$A^k u = ((x_1^T u)x_1 + (x_2^T u)x_2)\lambda_1^k + (x_3^T u)x_3\lambda_3^k + \cdots + (x_n^T u)x_n\lambda_n^k$$

and everything happens as if the eigenvalue λ_1 was simple and the matrix X had a larger nullspace. This is not a problem in the optimization problems discussed in this paper, since we need only *one* eigenvector in the leading invariant subspace, not the entire eigenspace.

Clustered eigenvalues (i.e. a small gap between the leading eigenvalue and the next one, not counting multiplicities) are much more problematic. The convergence of Ritz vectors cannot be established by the classical Chebyshev bounds described above, and various references provide a more refined analysis of this scenario (see Parlett, Simon and Stringer (1982), Van der Sluis and Van der Vorst (1987), Kuczynski and Wozniakowski (1992) among others). Successful termination of a *deterministic* Lanczos method can never be guaranteed anyway, since in the extreme case where the starting vector is orthogonal to the leading eigenspace, the Krylov subspace contains no information about leading eigenpairs. In practice, Lanczos solvers use *random* initial points. In particular, (Kuczynski and Wozniakowski, 1992, Th.4.2) show that, for any matrix $X \in \mathbf{S}_n$ (including matrices with clustered spectrum), starting the algorithm at a random u_1 picked uniformly over the sphere means the Lanczos decomposition will produce a leading Ritz pair with *relative* precision ϵ in

$$k^{\text{Lan}} \leq \frac{\log(n/\delta^2)}{4\sqrt{\epsilon}}$$

iterations, with probability at least $1-\delta$. This is of course a highly conservative bound and in particular, the worst case matrices used to prove it vary with k^{Lan}.

This means that computing one leading eigenpair of the matrix X requires computing at most k^{Lan} matrix vector products (we can always restart the code in case of failure) plus $4nk^{\text{Lan}}$ flops. When the matrix is dense, each

matrix vector product costs n^2 flops, hence the total cost of computing one leading eigenpair of X is

$$O\left(\frac{n^2\log(n/\delta^2)}{4\sqrt{\epsilon}}\right)$$

flops. When the matrix is sparse, the cost of each matrix vector product is $O(s)$ instead of $O(n^2)$, where s is the number of nonzero coefficients in X. Idem when the matrix X has rank $r < n$ and an explicit factorization is known (which is the case in the algorithms detailed in the previous section), in which case each matrix vector product costs $O(nr)$ which is the cost of two n by r matrix vector products, and the complexity of the Lanczos procedure decreases accordingly.

The numerical package ARPACK by Lehoucq, Sorensen and Yang (1998) implements the Lanczos procedure with a reverse communication interface allowing the user to efficiently compute the matrix vector product Xu_j. However, it uses the implicitly shifted QR method instead of the more efficient MRRR algorithm to compute the Ritz pairs of the matrix $T_k \in \mathbf{S}_k$.

5.2. *Other sampling techniques.* For completeness, we recall below another subsampling procedure in Achlioptas and McSherry (2007). More recent "volume sampling" techniques produce improved error bounds (some with multiplicative error bounds) but the corresponding optimal sampling probabilities are much harder to compute, we refer the reader to Kannan and Vempala (2009) for more details. The key idea behind this result is that, as the matrix dimension n grows and given a fixed, scale invariant precision target $\|X\|_F/\epsilon$, the norm $\|X\|_\infty$ of individual coefficients in X typically becomes negligible and we can randomly discard the majority of them while keeping important spectral features of X mostly intact.

LEMMA 8. *Given $X \in \mathbf{S}_n$ and $\epsilon > 0$, we define a subsampled matrix S whose coefficients are independently distributed as:*

$$S_{ij} = \begin{cases} X_{ij}/p & \text{with probability } p, \\ 0 & \text{otherwise.} \end{cases} \tag{21}$$

when $i \geq j$, and $S_{ij} = S_{ji}$ otherwise. Assume that $1 \geq p \geq (8\log n)^4/n$, then

$$\|X - S\|_2 \leq 4\|X\|_\infty\sqrt{n/p}.$$

with probability at least $1 - \exp(-19(\log n)^4)$.

PROOF. See (Achlioptas and McSherry, 2007, Th. 1.4). □

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

At first sight here, bounding the approximation error means letting the probability p grow relatively fast as n tends to infinity. However, because $\|X\|_\infty/\epsilon$ is typically much smaller than $\|X\|_F/\epsilon$, this subsampling ratio p can often be controlled. Adaptive subsampling, i.e. letting p vary with the magnitude of the coefficients in X, can further improve these results (see (Achlioptas and McSherry, 2007, §4) for details). The average number of nonzero coefficients in the subsampled matrix can be bounded using the structure of X. Note that the constants in this result are all very large (in particular, $1 \geq p \geq (8 \log n)^4/n$ implies $n \geq 10^9$) so despite its good empirical performance in low dimensions, the result presented above has to be understood in an asymptotic sense.

Acknowledgements. The author is grateful to two anonymous referees for their numerous comments, and would like to acknowledge partial support from NSF grants SES-0835550 (CDI), CMMI-0844795 (CAREER), CMMI-0968842, a starting grant from the European Research Council (ERC project SIPA), a Peek junior faculty fellowship, a Howard B. Wentz Jr. award and a gift from Google.

REFERENCES

Achlioptas, D. and McSherry, F. (2007). Fast computation of low-rank matrix approximations. *Journal of the ACM* **54**. MR2295993

Alon, A., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D. and Levine, A. J. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Cell Biology* **96** 6745–6750.

Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A. et al. (1999). *LAPACK Users' guide*. Society for Industrial Mathematics.

Arora, S. and Kale, S. (2007). A combinatorial, primal-dual approach to semidefinite programs In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing* 227–236. MR2402446

Bach, F. (2007). Consistency of trace norm minimization. *To appear in Journal of Machine Learning Research*. Arxiv preprint arXiv:0710.2848. MR2417263

Burke, J., Lewis, A. and Overton, M. (2002). Approximating Subdifferentials by Random Sampling of Gradients. *Mathematics of Operations Research* **27** 567–584. MR1926659

Burke, J. V., Lewis, A. S. and Overton, M. L. (2005). A Robust Gradient Sampling Algorithm for Nonsmooth, Nonconvex Optimization. *SIAM Journal on Optimization* **15** 751–779. MR2142859

Candes, E. J. and Recht, B. (2008). Exact matrix completion via convex optimization. *preprint*. MR2565240

d'Aspremont, A., Banerjee, O. and El Ghaoui, L. (2006). First-order methods for sparse covariance selection. *SIAM Journal on Matrix Analysis and Applications* **30** 56-66. MR2399568

d'Aspremont, A., El Ghaoui, L., Jordan, M. I. and Lanckriet, G. R. G. (2007). A Direct Formulation for Sparse PCA Using Semidefinite Programming. *SIAM Review* **49** 434-448. MR2353806

Dhillon, I. S. and Parlett, B. N. (2003). Orthogonal Eigenvectors and Relative Gaps. *SIAM Journal on Matrix Analysis and Applications* **25** 858-899. MR2081239

Dhillon, I. S., Parlett, B. N. and Vömel, C. (2006). The design and implementation of the MRRR algorithm. *ACM Transactions on Mathematical Software (TOMS)* **32** 560. MR2290445

Diaconis, P. (2003). Patterns in Eigenvalues: The 70th Josiah Willard Gibbs Lecture. *Bulletin of the American Mathematical Society* **40** 155-178. MR1962294

Drineas, P., Kannan, R. and Mahoney, M. W. (2006). Fast Monte Carlo Algorithms for Matrices II: Computing a Low-Rank Approximation to a Matrix. *SIAM Journal on Computing* **36** 158. MR2231644

Drineas, P., Kannan, R. and Mahoney, M. W. (2007). Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM Journal on Computing* **36** 132–157. MR2231643

El Karoui, N. (2007). Tracy-Widom limit for the largest eigenvalue of a large class of complex sample covariance matrices. *Annals of Probability* **35** 663–714. MR2308592

Fazel, M., Hindi, H. and Boyd, S. (2001). A rank minimization heuristic with application to minimum order system approximation. *Proceedings American Control Conference* **6** 4734-4739.

Frieze, A., Kannan, R. and Vempala, S. (2004). Fast monte-carlo algorithms for finding low-rank approximations. *Journal of the ACM (JACM)* **51** 1025–1041. MR2145262

Goemans, M. X. and Williamson, D. P. (1995). Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming. *J. ACM* **42** 1115-1145. MR1412228

Golub, G. H. and Van Loan, C. F. (1990). Matrix Computation. *North Oxford Academic*.

Horn, R. A. and Johnson, C. R. (1991). *Topics in matrix analysis*. Cambridge university press. MR1091716

Johnstone, I. M. (2001). On the distribution of the largest eigenvalue in principal components analysis. *Annals of Statistics* 295–327. MR1863961

Juditsky, A., Nemirovskii, A. S. and Tauvel, C. (2008). Solving variational inequalities with Stochastic Mirror-Prox algorithm. Arxiv preprint arXiv:0809.0815.

Juditsky, A., Lan, G., Nemirovski, A. and Shapiro, A. (2009). Stochastic Approximation Approach to Stochastic Programming. *SIAM Journal on Optimization* **19** 1574-1609. MR2486041

Kannan, R. and Vempala, S. (2009). *Spectral algorithms*. MR2558901

Kuczynski, J. and Wozniakowski, H. (1992). Estimating the largest eigenvalue by the power and Lanczos algorithms with a random start. *SIAM J. Matrix Anal. Appl* **13** 1094–1122. MR1182715

Kumar, K., Bhattacharya, C. and Hariharan, R. (2008). A Randomized Algorithm for Large Scale Support Vector Learning. In *Advances in Neural Information Processing Systems 20* (J. C. Platt, D. Koller, Y. Singer and S. Roweis, eds.) MIT Press, Cambridge, MA.

Lan, G. (2009). An optimal method for stochastic composite optimization. *Technical report, School of Industrial and Systems Engineering, Georgia Institute of Technology, 2009*.

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.

Lanckriet, G. R. G., Cristianini, N., Bartlett, P., El Ghaoui, L. and Jordan, M. I. (2002). Learning the Kernel Matrix with Semi-Definite Programming. *19th International Conference on Machine Learning*.

Lehoucq, R. B., Sorensen, D. C. and Yang, C. (1998). *ARPACK: Solution of Large-scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. Society for Industrial & Applied Mathematics. MR1621681

Nemirovsky, A. and Yudin, D. (1983). *Problem complexity and method efficiency in optimization*. MR0702836

Nesterov, Y. (2007). Gradient methods for minimizing composite objective function. *CORE DP2007/96*.

Nesterov, Y. (2009). Primal-dual subgradient methods for convex problems. *Mathematical programming Series B* **120** 221-259. MR2496434

Parlett, B., Simon, H. and Stringer, L. (1982). On estimating the largest eigenvalue with the Lanczos algorithm. *Mathematics of Computation* **38** 153–165. MR0637293

Polyak, B. and Juditsky, A. (1992). Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization* **30** 838. MR1167814

Polyak, B. T. and Shcherbakov, P. S. (2007). A RANDOMIZED METHOD FOR SOLVING SEMIDEFINITE PROGRAMS. *9th IFAC Workshop on Adaptation and Learning in Control and Signal Processing*.

Recht, B., Fazel, M. and Parrilo, P. A. (2007). Guaranteed Minimum-Rank Solutions of Linear Matrix Equations via Nuclear Norm Minimization. Arxiv preprint arXiv:0706.4138. MR2680543

Rudelson, M. and Vershynin, R. (2007). Sampling from large matrices: An approach through geometric functional analysis. *J. ACM* **54** 21. MR2351844

Saad, Y. (1992). *Numerical methods for large eigenvalue problems*. Manchester Univ Press. MR1177405

Srebro, N. (2004). Learning with Matrix Factorization PhD thesis, Massachusetts Institute of Technology. MR2717223

Stewart, G. W. (2001). *Matrix Algorithms Vol. II: Eigensystems*. Society for Industrial Mathematics. MR1853468

Stewart, G. W. and Sun, J. (1990). Matrix perturbation theory. MR1061154

Sun, J., Boyd, S., Xiao, L. and Diaconis, P. (2006). The fastest mixing Markov process on a graph and a connection to a maximum variance unfolding problem. *SIAM Review* **48** 681. MR2278445

Van der Sluis, A. and Van der Vorst, H. (1987). The convergence behavior of Ritz values in the presence of close eigenvalues. *Linear Algebra and its Applications* **88** 651–694. MR0882466

Weinberger, K. Q. and Saul, L. K. (2006). Unsupervised Learning of Image Manifolds by Semidefinite Programming. *International Journal of Computer Vision* **70** 77–90.

CMAP, École Polytechnique
UMR CNRS 7641
91128 Palaiseau, France
E-mail: alexandre.daspremont@m4x.org

Downloaded from informs.org by [216.73.217.172] on 15 November 2026, at 06:43 . For personal use only, all rights reserved.