# Semidefinite Programming on a Shoestring

**Alexandre d'Aspremont**, *CNRS & Ecole Polytechnique*.

Joint work with **Noureddine El Karoui**, *U.C. Berkeley*.

Support from NSF, ERC and Google.

# Introduction

Focus on **maximum eigenvalue minimization**

$$\min_{x \in Q} \lambda_{\max}\left(A_0 + \sum_{i=1}^{m} x_i A_i\right) + c^T x$$

in the variable $x \in \mathbb{R}^m$, with $A_i \in \mathbf{S}_n$, $c \in \mathbb{R}^m$.

- The set $Q$ is convex and simple, i.e. projections on $Q$ can be computed with low complexity.
- We also implicitly assume that $n$ is large while the target precision $\epsilon$ and the cost of forming $A(x) = A_0 + \sum_{i=1}^{m} x_i A_i$ remain relatively modest (e.g. $A_i$ sparse).

# Introduction

- All semidefinite programs with constant trace can be expressed in this way.
- In particular, many semidefinite relaxations of combinatorial problems fall in this setting (large $n$, modest precision target).
- The objective is non differentiable but can be regularized (more later).

# Introduction

Solve

$$\min_{x \in Q} \lambda_{\max}(A(x)) + c^T x$$

using **projected subgradient.**

**Input:** A starting point $x_0 \in \mathbb{R}^m$.

1: **for** $t = 0$ to $N-1$ **do**

2: Set

$$x_{t+1} = P_Q(x_t - \gamma \partial \lambda_{\max}(A(x))).$$

3: **end for**

**Output:** A point $x = (1/N)\sum_{t=1}^{N} x_t$.

- Here, $\gamma > 0$ and $P_Q(\cdot)$ is the Euclidean projection on $Q$.
- The number of iterations required to reach a target precision $\epsilon$ is

$$N = \frac{D_Q^2 M^2}{\epsilon^2}$$

where $D_Q$ is the diameter of $Q$ and $\|\partial \lambda_{\max}(A(x))\| \leq M$ on $Q$.

# Introduction

The **cost per iteration** is the sum of

- The cost $p_Q$ of computing the Euclidean projection on $Q$.
- The cost of computing $\partial\lambda_{\max}(A(x))$ which is e.g. $v_1 v_1^T$ where $v_1$ is a leading eigenvector of $X$.

Computing one leading eigenvector of a dense matrix $X$ with relative precision $\epsilon$, using a randomly started Lanczos method, with probability of failure $1-\delta$, costs

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

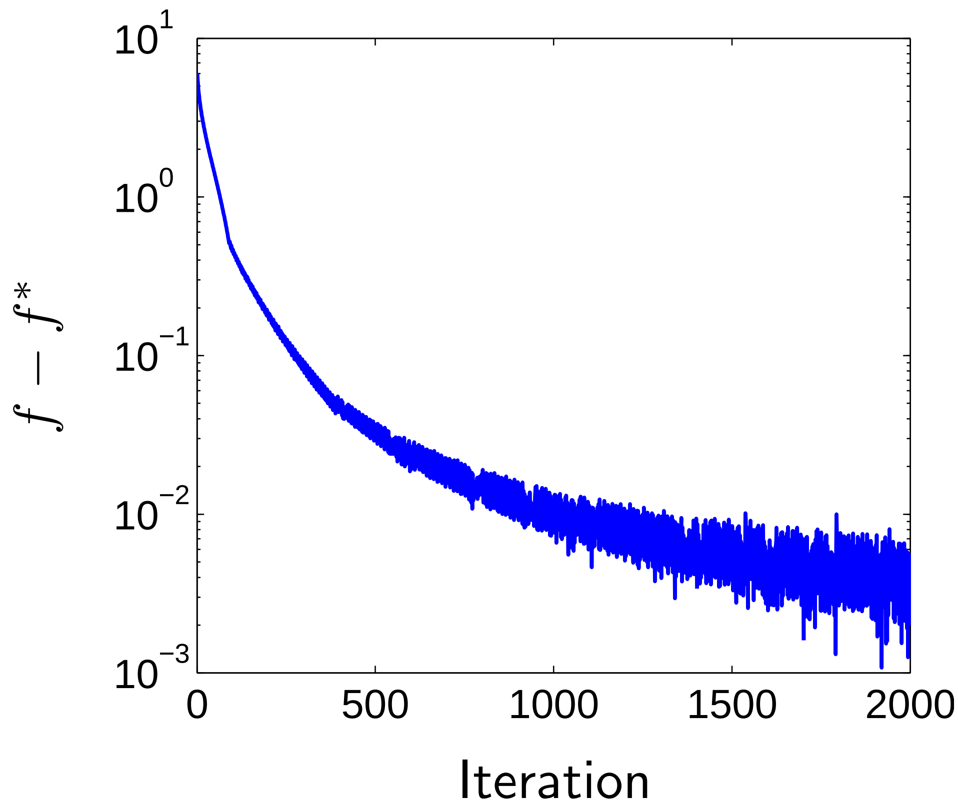
flops [Kuczynski and Wozniakowski, 1992, Th.4.2].

# Introduction

Solving $\min_{X \in Q} \lambda_{\max}(A(x))$ using projected subgradient.

- Easy to implement.
- Very poor performance in practice. The $1/\epsilon^2$ dependence is somewhat punishing. . .

Example below on MAXCUT.



# Introduction

[Nesterov, 2007a] We can **regularize** the objective and solve

$$\min_{x \in Q} \; f_\mu(x) \triangleq \mu \log \mathbf{Tr}\left(\exp\left(\frac{A(x)}{\mu}\right)\right)$$

for some regularization parameter $\mu > 0$ ($\exp(\cdot)$ is the **matrix** exponential here).

- If we set $\mu = \epsilon / \log n$ we get

$$\lambda_{\max}(A(x)) \leq f_\mu(x) \leq \lambda_{\max}(A(x)) + \epsilon$$

- The gradient $\nabla f_\mu(x)$ is Lipschitz continuous with constant

$$\frac{\|A\|^2 \log n}{\epsilon}$$

where $\|A\| = \sup_{\|h\| \leq 1} \|A(h)\|_2$.

# Introduction

- The number of iterations required to get an $\epsilon$ solution using the **smooth** minimization algorithm in Nesterov [1983] grows as

$$\frac{\|A\|\sqrt{\log n}}{\epsilon}\sqrt{\frac{d(x^*)}{\sigma}}$$

  where $d(\cdot)$ is strongly convex with parameter $\sigma > 0$.

- The **cost per iteration** is (usually) dominated by the cost of forming the matrix exponential

$$\exp\left(\frac{A(x)}{\mu}\right)$$

  which is $O(n^3)$ flops [Moler and Van Loan, 2003].

- Much better empirical performance.

# Introduction

This means that the two classical complexity options for solving

$$\min_{X \in Q} \lambda_{\max}(A(x))$$

(assuming $A(x)$ cheap)

- **Subgradient methods**

$$O\left(\frac{D_Q^2(n^2 \log n + p_Q)}{\epsilon^2}\right)$$

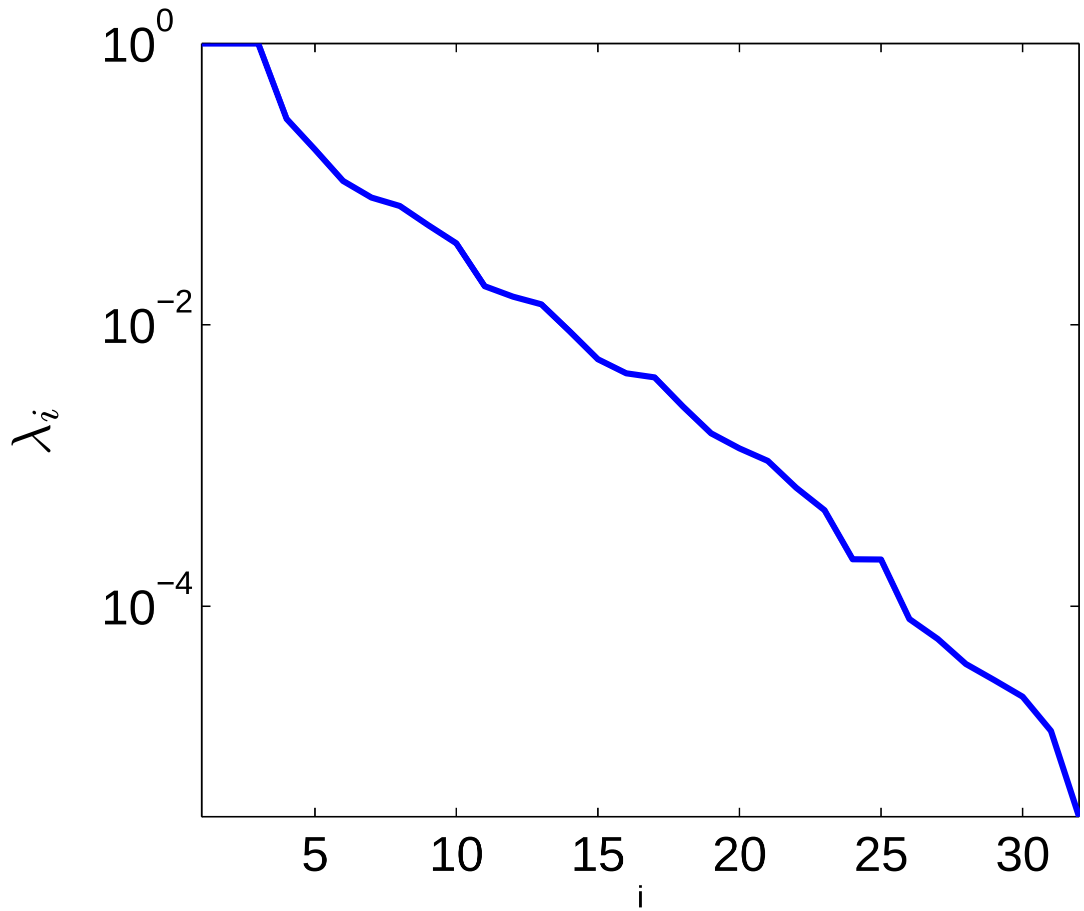
- **Smooth optimization**

$$O\left(\frac{D_Q\sqrt{\log n}(n^3 + p_Q)}{\epsilon}\right)$$

  if we pick $\|\cdot\|_2^2$ in the prox term.

# Introduction

Approximate gradient is often enough. This means computing only a few leading eigenvectors.




Spectrum of $\exp((X - \lambda_{\max}(X)\mathbf{I})/0.1)$ at the MAXCUT solution.

# Introduction

[d'Aspremont, 2008] Convergence guarantees using **approximate gradients**.

If $\tilde{\nabla} f(x)$ is the approximate gradient, we require

$$|\langle \tilde{\nabla} f(x) - \nabla f(x), y - z \rangle| \leq \delta \quad x, y, z \in Q,$$

hence the condition depends on the diameter of $Q$. For example, to solve

$$\begin{array}{ll} \mbox{minimize} & \lambda_{\max}(A + X) \\ \mbox{subject to} & |X_{ij}| \leq \rho \end{array}$$

we only compute the $j$ largest eigenvalues of $A + X$, with $j$ such that

$$\frac{(n-j)e^{\lambda_j}\sqrt{\sum_{i=1}^{j} e^{2\lambda_i}}}{(\sum_{i=1}^{j} e^{\lambda_i})^2} + \frac{\sqrt{n-j}\; e^{\lambda_j}}{\sum_{i=1}^{j} e^{\lambda_i}} \leq \frac{\delta}{\rho n}.$$

The impact of the **diameter** makes these conditions quite conservative.

# Introduction

Other conditions (often less stringent) are detailed in [Devolder, Glineur, and Nesterov, 2011] when solving

$$\min_{x\in Q}\ \max_{u\in U}\Psi(x,u)$$

If $u_x$ is an approximate solution to $\max_{u\in U}\Psi(x,u)$, we can check $V_i(u_x)\leq\delta$

$$V_1(u_x) = \max_{u\in U}\nabla_2\Psi(x,u_x)^T(u-u_x)$$

$$V_2(u_x) = \max_{u\in U}\left\{\Psi(x,u)-\Psi(x,u_x)+\kappa\|u-u_x\|^2/2\right\}$$

$$V_3(u_x) = \max_{u\in U}\Psi(x,u)-\Psi(x,u_x)$$

where

$$V_1(u_x)\leq V_2(u_x)\leq V_3(u_x)\leq\delta$$

- The target accuracy $\delta$ on the oracle is a function of the target accuracy $\epsilon$.
- Not clear yet if they can be tested independently of the diameter.

# Introduction

- Approximate gradients reduce empirical complexity. No **a priori** bounds on iteration cost.
- More efficient to run a lot of cheaper iterations, everything else being equal.

**Objectives**

- Keep some of the performance of smooth methods, while lowering the cost of smoothing?
- Get a more refined understanding of the iteration complexity versus convergence speed tradeoff?

One possible solution here: **stochastic gradient approximations.**

# Outline

- Introduction
- **Stochastic Smoothing**
- Maximum Eigenvalue Minimization

# Stochastic Smoothing

**Gaussian smoothing.** Suppose $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$ is Lipschitz continuous w.r.t. the Euclidean norm, with constant $\mu$. The function

$$g(x) = \mathbf{E}[f(x + (\sigma/\sqrt{n})u)]$$

where $u \sim \mathcal{N}(0, \mathbf{I}_n)$ and $\sigma > 0$, has a Lipschitz continuous gradient with

$$\|\nabla g(x) - \nabla g(y)\| \leq \frac{2\mu n}{\sigma}\|x - y\|.$$

Used in e.g. [Nesterov, 2011] to get explicit complexity bounds on gradient free optimization methods.

- $g(X) = \mathbf{E}[\lambda_{\max}(X + (\sigma/n)U]$ where $U \in \mathbf{S}_n$ is a symmetric matrix with standard normal upper triangle coefficients, has a Lipschitz continuous gradient with constant
$$O\left(\frac{n^3}{\sigma}\right)$$
- A smooth algorithm (if implementable) would require $O(n^{3/2})$ iterations.

# Stochastic Smoothing

**Gradient smoothness.** Call $f(X) = \lambda_{\max}(X)$, define

$$g(X,Y) = \lim_{t \to 0} \frac{\partial^2 f(X+tY)}{t^2}$$

and $L_f > 0$ such that

$$\|\nabla f(X) - \nabla g(Y)\| \leq L_f \|X - Y\|$$

we have

$$L_f = \sup_{X,Y} g(X,Y) = \sup_{X} \frac{1}{2(\lambda_1(X) - \lambda_2(X))}$$

The **spectral gap** controls the gradient's smoothness.

# Stochastic Smoothing

**Rank one updates.** Suppose $D \in \mathbf{S}_n$, we have almost explicit expressions for the eigenvalue decomposition of the matrix

$$X + \sigma u u^T$$

where $u \in \mathbb{R}^n$ and $\sigma > 0$.

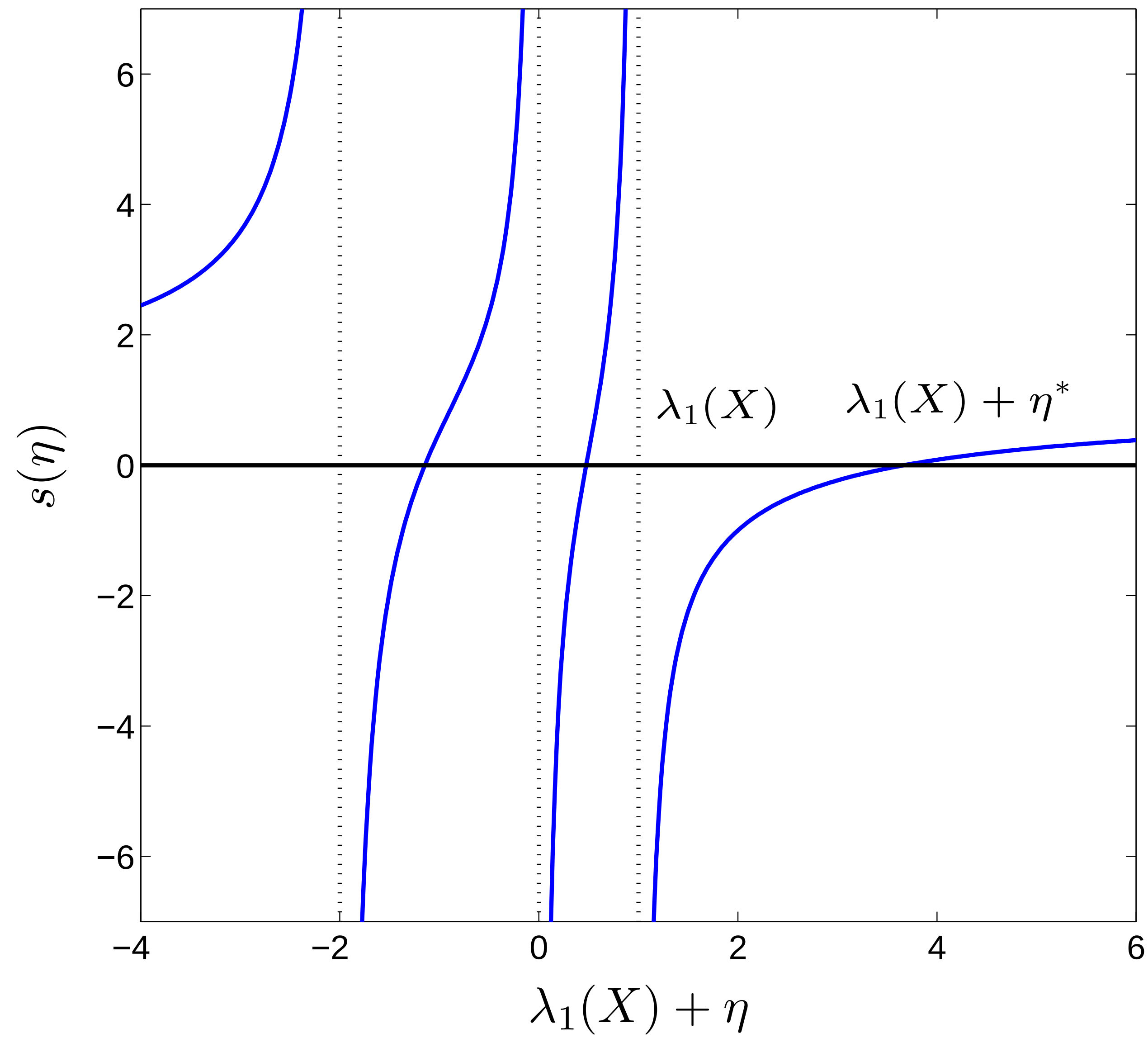
- W.l.o.g. we can assume $D$ is diagonal (just change $u$).
- If we write $\lambda_1(X + \sigma u u^T) = \lambda_1(X) + \eta$, we know that

$$\eta > 0 \quad \text{if } u_i \neq 0 \text{ for } i = 1, \ldots, n$$

- The eigenvalues of $X$ and $X + \sigma u u^T$ are **interlaced.**
- The increment $\eta^*$ is the unique positive root of the **secular equation**

$$s(\eta) \triangleq \frac{1}{\sigma} - \frac{u_1^2}{\eta} - \sum_{i=2}^{n} \frac{u_i^2}{(\lambda_1(X) - \lambda_i(X)) + \eta} = 0$$

# Stochastic Smoothing




Spectrum of $X$ is $\{-2,-2,0,1\}$, fourth eigenvalue of $X+\sigma uu^T$ at -2.

# Stochastic Smoothing

- The function
$$s^+(\eta) \triangleq \frac{1}{\sigma} - \frac{u_1^2}{\eta}$$
is an upper bound on $s(\eta)$.
- This means that the root of $s^+(\eta)$ is a **lower bound** on $\eta^*$ and we get
$$\eta^* \geq \frac{u_1^2}{\sigma}$$
- Together with interlacing, this yields
$$\lambda_2(X + \sigma uu^T) \leq \lambda_1(X) \leq \lambda_1(X) + \eta^* \leq \lambda_1(X + \sigma uu^T)$$

Finally, we get **a lower bound on the spectral gap**

$$\lambda_1(X + \sigma uu^T) - \lambda_2(X + \sigma uu^T) \geq \frac{u_1^2}{\sigma}$$

# Stochastic Smoothing

**Rank one Gaussian smoothing.** Suppose we pick $u \in \mathbb{R}^n$ with i.i.d. $u_i \sim \mathcal{N}(0,1)$ and define

$$f(X) = \mathbf{E}[\lambda_{\max}(X + (\epsilon/n)uu^T)]$$

for some $\epsilon > 0$.

- Because $uu^T \succeq 0$ and $\lambda(\cdot)$ is 1-Lipschitz

$$\lambda_{\max}(X) \leq \mathbf{E}[\lambda_{\max}(X + (\epsilon/n)uu^T)] \leq \lambda_{\max}(X) + \epsilon$$

- The Gaussian distribution is rotationally invariant, so the spectral gap is bounded below by

$$\frac{\epsilon\, u_1^2}{n}$$

where $u_1 \sim \mathcal{N}(0,1)$.

Unfortunately $\mathbf{E}[1/u_1^2] = +\infty$, easy to fix. . .

# Stochastic Smoothing

**Max-rank one Gaussian smoothing.** Suppose we pick $u_i \in \mathbb{R}^n$ with i.i.d. $u_{ij} \sim \mathcal{N}(0,1)$ and define

$$f(X) = \mathbf{E}\left[\max_{i=1,\ldots,k} \lambda_{\max}(X + (\epsilon/n)u_i u_i^T)\right]$$

- Approximation results are preserved up to a constant $c_k > 0$

$$\lambda_{\max}(X) \leq \mathbf{E}[\lambda_{\max}(X + (\epsilon/n)uu^T)] \leq \lambda_{\max}(X) + c_k\epsilon$$

- The Gaussian distribution is rotationally invariant, so the spectral gap is bounded below by

$$\max_{i=1,\ldots,k} \frac{\epsilon\ u_{i,1}^2}{n}$$

  where $u_i$ are i.i.d. with $u_{i,1} \sim \mathcal{N}(0,1)$.
- The complexity of computing $\max_{i=1,\ldots,k} \lambda_{\max}(X + (\epsilon/n)u_i u_i^T)$ is

$$O(kn^2 \log n).$$

# Stochastic Smoothing

**Proposition 1**

**Max-rank one Gaussian smoothing.** *The function*

$$f(X) = \mathbf{E}\left[\max_{i=1,\ldots,k} \lambda_{\max}(X + (\epsilon/n)u_i u_i^T)\right]$$

*is smooth and the Lipschitz constant of its gradient is bounded by*

$$L_f \leq \mathbf{E}\left[\frac{n}{2\epsilon}\left(\min_{i=1,\ldots,k} \frac{1}{u_{i,1}^2}\right)\right] \leq C_k \frac{n}{\epsilon}$$

*where $C_k < \infty$ when $k \geq 3$.*

# Stochastic Smoothing

**Gradient variance.** We have

$$\partial\lambda_{\max}(X) = v_1(X)v_1(X)^T$$

where $v_1(X)$ is a leading eigenvector of $X$.

- We have, when $D$ is diagonal

$$v_1(D+uu^T)_i = c\frac{u_i}{\lambda_1(D+uu^T)-\lambda_i(D)}$$

  where $c>0$ is a normalization term.

- By symmetry, when $u$ is Gaussian, $A=\mathbf{E}[v_1(X+uu^T)v_1(X+uu^T)^T]$ is diagonal, with

$$\mathbf{E}[\mathbf{Tr}\left(v_1v_1^T-A\right)^2]=1-\mathbf{Tr}\,A^2,$$

  where $\mathbf{Tr}\,A=1$ with $A_{ii}\geq 0$.

This means that $\mathbf{E}[\mathbf{Tr}\left(v_1v_1^T-A\right)^2]$ is of order 1.

# Outline

- Introduction
- Stochastic Smoothing
- **Maximum Eigenvalue Minimization**

# Maximum Eigenvalue Minimization

Solve **maximum eigenvalue minimization** after stochastic smoothing

$$\min_{x \in Q} \mathbf{E} \left[ \max_{j=1,\ldots,3} \lambda_{\max} \left( A_0 + \sum_{i=1}^{m} x_i A_i + \frac{\epsilon}{n} u_j u_j^T \right) \right] + c^T x$$

in the variable $x \in \mathbb{R}^m$, with $A_i \in \mathbf{S}_n$, $c \in \mathbb{R}^m$ and the $u_j$ are Gaussian.

We use an optimal stochastic minimization algorithm in [Lan, 2009] which is a generalization of the algorithm in Nesterov [1983].

# Maximum Eigenvalue Minimization

**Optimal Stochastic Composite Optimization.** The algorithm in Lan [2009] solves

$$\min_{x\in Q} \Psi(x) \triangleq f(x) + h(x)$$

with the following assumptions

- $f(x)$ has Lipschitz gradient with constant $L$ and $h(x)$ is Lipschitz with constant $M$,
- we have a **stochastic oracle** $G(x, \xi_t)$ for the gradient, which satisfies

$$\mathbf{E}[G(x,\xi_t)] = g(x) \in \partial\Psi(x) \quad \text{and} \quad \mathbf{E}[\|G(x,\xi_t) - g(x)\|_*^2] \leq \sigma^2$$

After $N$ iterations, the iterate $x_{N+1}$ satisfies

$$\mathbf{E}\left[\Psi(x_{N+1}^{ag}) - \Psi^*\right] \leq \frac{8LD_{\omega,Q}^2}{N^2} + \frac{4D_{\omega,Q}\sqrt{4\mathcal{M}^2+\sigma^2}}{\sqrt{N}}$$

which is optimal. Additional assumptions guarantee convergence w.h.p.

# Maximum Eigenvalue Minimization

**Stochastic line search.**

- The bounds on variance and smoothness are very conservative.
- Line search allows to take full advantage of the smoothness of $\lambda_{\max}(X)$ outside of pathological areas.

**Monotonic line search.** In Lan [2009], we test

$$\begin{aligned}\Psi(x_{t+1}^{ag}, \xi_{t+1}) &\leq \Psi(x_t^{md}, \xi_t) + \langle G(x_t^{md}, \xi_t), x_{t+1}^{ag} - x_t^{md} \rangle \\ &\quad + \frac{\alpha}{4\gamma_t \beta_t} \|x_{t+1}^{ag} - x_t^{md}\|^2 + 2\mathcal{M}\|x_{t+1}^{ag} - x_t^{md}\|\end{aligned}$$

while decreasing the step size monotonically across iterations.

## Optimal Smooth Stochastic Minimization with Line Search.

**Input:** An initial point $x^{ag} = x_1 = x^w \in \mathbb{R}^n$, an iteration counter $t = 1$, the number of iterations $N$, line search parameters $\gamma^{min}, \gamma^{max}$, $\gamma^d$, $\gamma > 0$, with $\gamma^d < 1$.

1: Set $\gamma = \gamma^{max}$.
2: **for** $t = 1$ to $N$ **do**
3: Define $x_t^{md} = \frac{2}{t+1} x_t + \frac{t-1}{t+1} x_t^{ag}$
4: Call the stochastic gradient oracle to get $G(x_t^{md}, \xi_t)$.
5: **repeat**
6: Set $\gamma_t = \frac{(t+1)\gamma}{2}$.
7: Compute the prox mapping $x_{t+1} = P_{x_t}(\gamma_t G(x_t^{md}, \xi_t))$.
8: Set $x_{t+1}^{ag} = \frac{2}{t+1} x_{t+1} + \frac{t-1}{t+1} x_t^{ag}$.
9: **until** $\Psi(x_{t+1}^{ag}, \xi_{t+1}) \leq$ $\Psi(x_t^{md}, \xi_t) + \langle G(x_t^{md}, \xi_t), x_{t+1}^{ag} - x_t^{md} \rangle + \frac{\alpha \gamma^d}{4\gamma} \|x_{t+1}^{ag} - x_t^{md}\|^2 + 2\mathcal{M} \|x_{t+1}^{ag} - x_t^{md}\|$ or $\gamma \leq \gamma^{min}$. If exit condition fails, set $\gamma = \gamma\gamma^d$ and go back to step 5.
10: Set $\gamma = \max\left\{\gamma^{min}, \gamma\right\}$.
11: **end for**

**Output:** A point $x_{N+1}^{ag}$.

# Maximum Eigenvalue Minimization

For maximum eigenvalue minimization

- We have $\sigma \leq 1$, but we can reduce this by averaging $q$ gradients, to control the tradeoff between smooth and non-smooth terms.
- If we set $q = \max\{1, D_Q/(\epsilon\sqrt{n})\}$ and $N = 2D_Q\sqrt{n}/\epsilon$ we get the following complexity picture

| **Complexity** | Num. of Iterations | Cost per Iteration |
|---:|:---:|:---:|
| Nonsmooth alg. | $O\left(\frac{D_Q^2}{\epsilon^2}\right)$ | $O(p_Q + n^2 \log n)$ |
| Smooth stochastic alg. | $O\left(\frac{D_Q\sqrt{n}}{\epsilon}\right)$ | $O\left(p_Q + \max\left\{1, \frac{D_Q}{\epsilon\sqrt{n}}\right\} n^2 \log n\right)$ |
| Smoothing alg. | $O\left(\frac{D_Q\sqrt{\log n}}{\epsilon}\right)$ | $O(p_Q + n^3)$ |

# Conclusion

- Stochastic smoothing with a few eigenvalues.
- Explicit control of the iteration cost versus smoothness tradeoff.

Some open problems. . .

- Not clear how to get convergence with high probability.
- Stochastic algorithm with non monotonic step sizes?

*

References


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

A. d'Aspremont. Smooth optimization with approximate gradient. *SIAM Journal on Optimization*, 19(3):1171–1183, 2008.

O. Devolder, F. Glineur, and Y. Nesterov. First-order methods of smooth convex optimization with inexact oracle. *CORE Discussion Papers,(2011/02)*, 2011.

J. Kuczynski and H. Wozniakowski. Estimating the largest eigenvalue by the power and Lanczos algorithms with a random start. *SIAM J. Matrix Anal. Appl*, 13(4):1094–1122, 1992.

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

C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.

Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2): 372–376, 1983.

Y. Nesterov. Smoothing technique and its applications in semidefinite optimization. *Mathematical Programming*, 110(2):245–259, 2007a.

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

Y. Nesterov. Random gradient-free minimization of convex functions. *CORE Discussion Papers*, 2011.