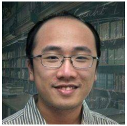
# Computing High-Precision Lewis Weights

Maryam Fazel [1], Yin Tat Lee [1], Swati Padmanabhan[1], and Aaron Sidford[2]

[1]University of Washington; [2]Stanford University

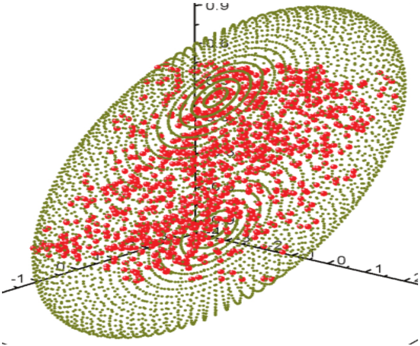



# Problem Statement

# Warmup: John Ellipsoid

# Warmup: John Ellipsoid

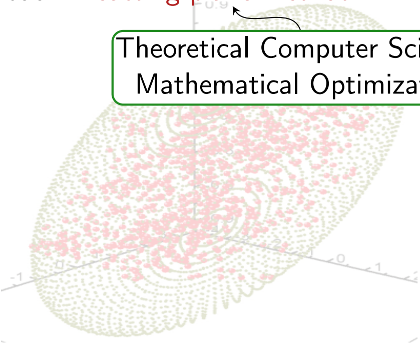
- Smallest ellipsoid enclosing a given set of points $\{a_i\}_{i=1}^m \in \mathbb{R}^n$



# Warmup: John Ellipsoid

- Smallest ellipsoid enclosing a given set of points $\{a_i\}_{i=1}^m \in \mathbb{R}^n$
- Algorithmic tool in cutting-plane method

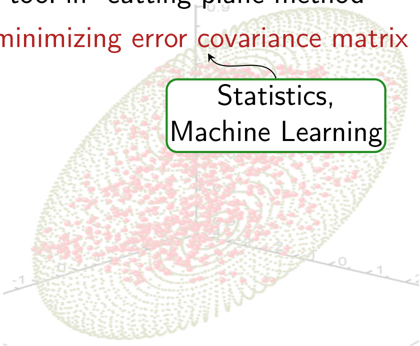
Theoretical Computer Science, Mathematical Optimization



# Warmup: John Ellipsoid

- Smallest ellipsoid enclosing a given set of points $\{a_i\}_{i=1}^m \in \mathbb{R}^n$
- Algorithmic tool in cutting-plane method
- Applied in minimizing error covariance matrix

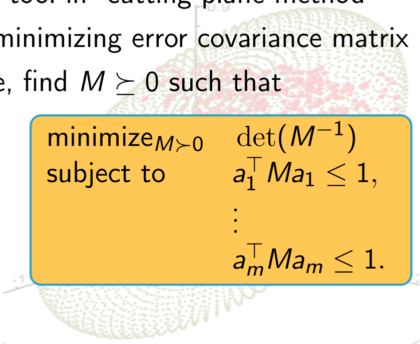
Statistics,
Machine Learning



# Warmup: John Ellipsoid

- Smallest ellipsoid enclosing a given set of points $\{a_i\}_{i=1}^m \in \mathbb{R}^n$
- Algorithmic tool in cutting-plane method
- Applied in minimizing error covariance matrix
- To compute, find $M \succeq 0$ such that

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & a_1^\top M a_1 \leq 1, \\ & \vdots \\ & a_m^\top M a_m \leq 1. \end{array}$$



# Warmup: John Ellipsoid

- Smallest ellipsoid enclosing a given set of points $\{a_i\}_{i=1}^m \in \mathbb{R}^n$
- Algorithmic tool in cutting-plane method
- Applied in minimizing error covariance matrix
- To compute, find $M \succeq 0$ such that

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & a_1^\top M a_1 \leq 1, \\ & \vdots \\ & a_m^\top M a_m \leq 1. \end{array}$$

- Optimal ellipsoid is $M^{-1} = \sum_{i=1}^m \overline{w}_i a_i a_i^\top$, where

$$a_i^\top (A^\top \mathrm{diag}(\overline{w}) A)^{-1} a_i = 1.$$

# More Generally: Lewis Ellipsoid

Find the Lewis Ellipsoid $M \succeq 0$ that satisfies

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & \sum_{i=1}^{m} (a_i^\top M a_i)^{p/2} \leq 1 \end{array} .$$

# More Generally: Lewis Ellipsoid

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & a_i^\top M a_i \leq 1, \forall i \end{array}$$

Find the Lewis Ellipsoid $M \succeq 0$ that satisfies

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & \sum_{i=1}^{m} (a_i^\top M a_i)^{p/2} \leq 1 \end{array} .$$

$p = \infty$ gives John Ellipsoid

# More Generally: Lewis Ellipsoid

Find the Lewis Ellipsoid $M \succeq 0$ that satisfies

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & \sum_{i=1}^{m} (a_i^\top M a_i)^{p/2} \leq 1 \end{array}.$$

$\overline{w} \in \mathbb{R}^m_{\geq 0}$ "Lewis weights"

- Lewis ellipsoid satisfies $M^{-1} = \sum_{i=1}^{m} \overline{w_i} a_i a_i^\top$, where

$$a_i^\top (A^\top \mathrm{diag}(\overline{w})^{1-2/p} A)^{-1} a_i = \overline{w}_i^{2/p}.$$

# More Generally: Lewis Ellipsoid

Find the Lewis Ellipsoid $M \succeq 0$ that satisfies

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & \sum_{i=1}^{m} (a_i^\top M a_i)^{p/2} \leq 1 \end{array}.$$

- Lewis ellipsoid satisfies $M^{-1} = \sum_{i=1}^{m} \overline{w}_i a_i a_i^\top$, where

$$a_i^\top (A^\top \text{diag}(\overline{w})^{1-2/p} A)^{-1} a_i = \overline{w}_i^{2/p}.$$

$\overline{w} \in \mathbb{R}^m_{\geq 0}$
"Lewis weights"

Existence proved by D.R. Lewis in 1978

# More Generally: Lewis Ellipsoid

Find the Lewis Ellipsoid $M \succeq 0$ that satisfies

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & \sum_{i=1}^m (a_i^\top M a_i)^{p/2} \leq 1 \end{array} .$$

- Lewis ellipsoid satisfies $M^{-1} = \sum_{i=1}^m \overline{w}_i a_i a_i^\top$, where

$$a_i^\top (A^\top \mathrm{diag}(\overline{w})^{1-2/p} A)^{-1} a_i = \overline{w}_i^{2/p}.$$

$\overline{w} \in \mathbb{R}^m_{\geq 0}$ "Lewis weights"

- Applications:

# More Generally: Lewis Ellipsoid

Find the Lewis Ellipsoid $M \succeq 0$ that satisfies

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & \sum_{i=1}^{m} (a_i^\top M a_i)^{p/2} \leq 1 \end{array}.$$

$\overline{w} \in \mathbb{R}^m_{\geq 0}$ "Lewis weights"

- Lewis ellipsoid satisfies $M^{-1} = \sum_{i=1}^{m} \overline{w}_i a_i a_i^\top$, where

$$a_i^\top (A^\top \text{diag}(\overline{w})^{1-2/p} A)^{-1} a_i = \overline{w}_i^{2/p}.$$

- Applications:
  - sampling important rows of data matrix,

Statistics, Machine Learning

# More Generally: Lewis Ellipsoid

Find the Lewis Ellipsoid $M \succeq 0$ that satisfies

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & \sum_{i=1}^{m} (a_i^\top M a_i)^{p/2} \leq 1 \end{array}.$$

$\overline{w} \in \mathbb{R}^m_{\geq 0}$ "Lewis weights"

- Lewis ellipsoid satisfies $M^{-1} = \sum_{i=1}^{m} \overline{w_i} a_i a_i^\top$, where

$$a_i^\top (A^\top \mathrm{diag}(\overline{w})^{1-2/p} A)^{-1} a_i = \overline{w}_i^{2/p}.$$

- Applications:
  - sampling important rows of data matrix,
  - computing the Lee-Sidford self-concordant barrier

Mathematical Optimization

# Optimization Perspective

Lewis weights are the solution to the convex program

$\alpha = \frac{2}{p-2}$

$$\operatorname{argmin}_{w \geq 0} \quad -\log\det(A^\top \operatorname{diag}(w) A) + \frac{1}{1+\alpha} 1^\top w^{1+\alpha}.$$

# Optimization Perspective

Lewis weights are the solution to the convex program

$$\mathrm{argmin}_{w\geq 0} \quad -\log\det(A^\top \mathrm{diag}(w)A) + \frac{1}{1+\alpha}1^\top w^{1+\alpha}.$$

- Optimality condition of the above program is

$$a_i^\top (A^\top \mathrm{diag}(\overline{w})^{1-2/p}A)^{-1}a_i = \overline{w}_i^{2/p}.$$

# Optimization Perspective

$$\text{minimize}_{M \succ 0} \quad \det(M^{-1})$$
$$\text{subject to} \quad \sum_{i=1}^{m} (a_i^\top M a_i)^{p/2} \leq 1.$$

Lewis weights are the solution to the convex program

$$\text{argmin}_{w \geq 0} \quad -\log \det(A^\top \mathrm{diag}(w) A) + \frac{1}{1+\alpha} 1^\top w^{1+\alpha}.$$

- Optimality condition of the above program is

$$a_i^\top (A^\top \mathrm{diag}(\overline{w})^{1-2/p} A)^{-1} a_i = \overline{w}_i^{2/p}.$$

# Our Goal

- A high precision algorithm to compute $w$ satisfying $w \approx_{\varepsilon} \overline{w}$

# Our Goal

$(1-\varepsilon)w_i \leq \overline{w}_i \leq (1+\varepsilon)w_i$

- A high precision algorithm to compute $w$ satisfying $w \approx_{\varepsilon} \overline{w}$

# Our Goal

Runtime $\operatorname{poly}\log(1/\varepsilon)$

$(1-\varepsilon)w_i \leq \overline{w}_i \leq (1+\varepsilon)w_i$

- A high precision algorithm to compute $w$ satisfying $w \approx_\varepsilon \overline{w}$

# Our Goal

Runtime $\mathrm{poly}\log(1/\varepsilon)$

$(1-\varepsilon)w_i \leq \overline{w}_i \leq (1+\varepsilon)w_i$

- A high precision algorithm to compute $w$ satisfying $w \approx_{\varepsilon} \overline{w}$
  - For many applications, approximate Lewis weights suffice!

sampling

# Our Goal

Runtime $\mathrm{poly}\log(1/\varepsilon)$

$(1-\varepsilon)w_i \leq \overline{w}_i \leq (1+\varepsilon)w_i$

- A high precision algorithm to compute $w$ satisfying $w \approx_{\varepsilon} \overline{w}$
  - For many applications, approximate Lewis weights suffice!
  - High-precision Lewis weights useful in computing volume of polytope.

# Our Goal

Runtime $\mathrm{poly}\log(1/\varepsilon)$

$(1-\varepsilon)w_i \le \overline{w}_i \le (1+\varepsilon)w_i$

- A high precision algorithm to compute $w$ satisfying $w \approx_\varepsilon \overline{w}$
  - For many applications, approximate Lewis weights suffice!
  - High-precision Lewis weights useful in computing volume of polytope.
- Runtime measured in number of leverage score computations

$\sigma_i(w) = w_i \cdot a_i^\top (A^\top \mathrm{diag}(w) A)^{-1} a_i$

# Our Goal

Runtime $\mathrm{poly}\log(1/\varepsilon)$

$(1-\varepsilon)w_i \leq \overline{w}_i \leq (1+\varepsilon)w_i$

- A high precision algorithm to compute $w$ satisfying $w \approx_\varepsilon \overline{w}$
  - For many applications, approximate Lewis weights suffice!
  - High-precision Lewis weights useful in computing volume of polytope.
- Runtime measured in number of leverage score computations
- Concretely:

Compute $\ell_p$ Lewis weights with $\mathrm{poly}\log(1/\varepsilon)$ leverage scores.

# Prior Work on Computing Lewis Weights

# Contrasting Our Results with Previous Work

| Year | Authors | Range of $p$ | # Leverage Score Computations | Total Run Time |
|---|---|---|---|---|
| 2015 | Cohen, Peng | $p \in (0, 4)$ | $O(\frac{1}{1-\lvert 1-p/2\rvert} \log(\frac{\log m}{\varepsilon}))$ | $O(\frac{1}{\lvert 1-p/2\rvert} mn^2 \log(\frac{\log m}{\varepsilon}))$ |
| 2015 | Cohen, Peng | $p \geq 4$ | $\Omega(n)$ | $\Omega(mn^3 \log\left(\frac{m}{\varepsilon}\right))$ |
| 2015 | Cohen, Peng | $p \geq 4$ | N/A | $O(\frac{\mathrm{nnz}(A)}{\varepsilon} + c_p n^{O(p)})$ |
| 2016 | Lee | $p \geq 4$ | $\frac{1}{\varepsilon} \log(m/n)$ | $O\left(\left(\frac{1}{\varepsilon}\mathrm{nnz}(A) + \frac{n^3}{\epsilon^3}\right) \log(m/n)\right)$ |
| 2019 | Lee, Sidford | $p \geq 4$ | $O(p^2\sqrt{n})$ | $O(p^2 mn^{2.5} \mathrm{polylog}(\frac{m}{\varepsilon}))$ |
| 2021 | Our result | $p \geq 4$ | $O(p^3 \log(\frac{m}{\varepsilon}))$ | $O(p^3 mn^2 \log(\frac{m}{\varepsilon}))$ |

- Previously: high-precision algorithm only for $0 \leq p < 4$

# Contrasting Our Results with Previous Work

| Year | Authors | Range of $p$ | # Leverage Score Computations | Total Run Time |
|---|---|---|---|---|
| 2015 | Cohen, Peng | $p \in (0, 4)$ | $O(\frac{1}{1-\lvert 1-p/2 \rvert} \log(\frac{\log m}{\varepsilon}))$ | $O(\frac{1}{\lvert 1-p/2 \rvert} mn^2 \log(\frac{\log m}{\varepsilon}))$ |
| 2015 | Cohen, Peng | $p \geq 4$ | $\Omega(n)$ | $\Omega(mn^3 \log\left(\frac{m}{\varepsilon}\right))$ |
| 2015 | Cohen, Peng | $p \geq 4$ | N/A | $O(\frac{\mathrm{nnz}(A)}{\varepsilon} + c_p n^{O(p)})$ |
| 2016 | Lee | $p \geq 4$ | $\frac{1}{\varepsilon} \log(m/n)$ | $O\left(\left(\frac{1}{\varepsilon}\mathrm{nnz}(A) + \frac{n^3}{\epsilon^3}\right) \log(m/n)\right)$ |
| 2019 | Lee, Sidford | $p \geq 4$ | $O(p^2\sqrt{n})$ | $O(p^2 mn^{2.5} \mathrm{polylog}(\frac{m}{\varepsilon}))$ |
| 2021 | Our result | $p \geq 4$ | $O(p^3 \log(\frac{m}{\varepsilon}))$ | $O(p^3 mn^2 \log(\frac{m}{\varepsilon}))$ |

- Previously: high-precision algorithm only for $0 \leq p < 4$
- We provide a high-precision algorithm for $p \geq 4$.

# Contrasting Our Results with Previous Work

| Year | Authors | Range of $p$ | # Leverage Score Computations | Total Run Time |
|---|---|---|---|---|
| 2015 | Cohen, Peng | $p \in (0, 4)$ | $O(\frac{1}{1-\lvert 1-p/2\rvert} \log(\frac{\log m}{\varepsilon}))$ | $O(\frac{1}{\lvert 1-p/2\rvert} mn^2 \log(\frac{\log m}{\varepsilon}))$ |
| 2015 | Cohen, Peng | $p \geq 4$ | $\Omega(n)$ | $\Omega(mn^3 \log\left(\frac{m}{\varepsilon}\right))$ |
| 2015 | Cohen, Peng | $p \geq 4$ | N/A | $O(\frac{\mathrm{nnz}(A)}{\varepsilon} + c_p n^{O(p)})$ |
| 2016 | Lee | $p \geq 4$ | $\frac{1}{\varepsilon} \log(m/n)$ | $O\left(\left(\frac{1}{\varepsilon}\mathrm{nnz}(A) + \frac{n^3}{\epsilon^3}\right) \log(m/n)\right)$ |
| 2019 | Lee, Sidford | $p \geq 4$ | $O(p^2\sqrt{n})$ | $O(p^2 mn^{2.5} \mathrm{polylog}(\frac{m}{\varepsilon}))$ |
| 2021 | Our result | $p \geq 4$ | $O(p^3 \log(\frac{m}{\varepsilon}))$ | $O(p^3 mn^2 \log(\frac{m}{\varepsilon}))$ |

- Previously: high-precision algorithm only for $0 \leq p < 4$
- We provide a high-precision algorithm for $p \geq 4$.
- Therefore, there now exists a high-precision algorithm for all $p > 0$.

# Prior Work: Cohen-Peng'15

- Repeatedly, $w_i \leftarrow (a_i^\top (A^\top \mathrm{diag}(w)^{1-2/p} A)^{-1} a_i)^{p/2}$ for all $i$.

# Prior Work: Cohen-Peng'15

- Repeatedly, $w_i \leftarrow (a_i^\top (A^\top \mathrm{diag}(w)^{1-2/p} A)^{-1} a_i)^{p/2}$ for all $i$.
- The above map is a contraction only for $p < 4$.

# Prior Work: Cohen-Peng'15

- Repeatedly, $w_i \leftarrow (a_i^\top (A^\top \mathrm{diag}(w)^{1-2/p} A)^{-1} a_i)^{p/2}$ for all $i$.
- The above map is a contraction only for $p < 4$.

For $p \geq 4$, solve the convex program

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & \sum_{i \in [m]} (a_i^\top M a_i)^{p/2} \leq 1. \end{array}$$

# Prior Work: Cohen-Peng'15

- Repeatedly, $w_i \leftarrow (a_i^\top (A^\top \mathrm{diag}(w)^{1-2/p} A)^{-1} a_i)^{p/2}$ for all $i$.
- The above map is a contraction only for $p < 4$.

For $p \geq 4$, solve the convex program

$$\begin{array}{ll} \text{minimize}_{M \succ 0} & \det(M^{-1}) \\ \text{subject to} & \sum_{i \in [m]} (a_i^\top M a_i)^{p/2} \leq 1. \end{array}$$

- Other approaches: mirror descent, homotopy method, ...

# Our Contributions

# Our Technical Outline

We compute Lewis weights by solving:

$$\text{argmin}_{w \geq 0} \quad -\log\det(A^\top \text{diag}(w) A) + \frac{1}{1+\alpha} 1^\top w^{1+\alpha}.$$

$\alpha = \frac{2}{p-2}$

# Our Technical Outline

We compute Lewis weights by solving:

$$\text{argmin}_{w \geq 0} \quad -\log \det(A^\top \text{diag}(w) A) + \frac{1}{1+\alpha} 1^\top w^{1+\alpha}.$$

$\alpha = \frac{2}{p-2}$

- Standard high-accuracy algorithms don't apply

# Our Technical Outline

We compute Lewis weights by solving:

$$\text{argmin}_{w \geq 0} \quad -\log \det(A^\top \text{diag}(w) A) + \frac{1}{1+\alpha} 1^\top w^{1+\alpha}.$$

$\alpha = \frac{2}{p-2}$

- Standard high-accuracy algorithms don't apply
- Identify a "rounding" condition that allows geometric decrease in error

# Our Technical Outline

We compute Lewis weights by solving:

$\alpha = \frac{2}{p-2}$

$$\mathrm{argmin}_{w \geq 0} \quad -\log \det(A^\top \mathrm{diag}(w) A) + \frac{1}{1+\alpha} 1^\top w^{1+\alpha}.$$

- Standard high-accuracy algorithms don't apply
- Identify a "rounding" condition that allows geometric decrease in error

Leverages upper & lower quadratics

# Our Technical Outline

We compute Lewis weights by solving:

$\alpha = \frac{2}{p-2}$

$$\mathrm{argmin}_{w \geq 0} \quad -\log \det(A^\top \mathrm{diag}(w) A) + \frac{1}{1+\alpha} 1^\top w^{1+\alpha}.$$

- Standard high-accuracy algorithms don't apply
- Identify a "rounding" condition that allows geometric decrease in error
- Maintain the rounding condition in constant number of steps

# Our Algorithm

1. Initialize $w^{(0)} = n/m$.
2. For $k = 1, 2, 3, \ldots, T$ iterations, do:
   - $\widetilde{w}^{(k)} \leftarrow \textbf{Round}(w^{(k-1)})$ // rounding condition
   - $w^{(k)} \leftarrow \textbf{Descent}(\widetilde{w}^{(k)}, [m], 1)$ //make progress
3. Set $w_R \leftarrow \textbf{Round}(w^T)$
4. Return $\widehat{w}$ where $\widehat{w}_i \leftarrow (a_i^\top (\mathrm{AW}_R\mathrm{A})^{-1} a_i)^{1/\alpha}$.

# Our First Technical Building Block: **Descent**

$$\rho_i(w) = \frac{a_i^\top (A^\top \mathrm{diag}(w) A)^{-1} a_i}{w_i^\alpha}$$

The descent step:

$$[\mathrm{Descent}(w, C, \eta)]_i := w_i \left[1 + \eta_i \cdot \frac{\rho_i(w) - 1}{\rho_i(w) + 1}\right].$$

for $i \in C \subseteq \{1, 2, \ldots, m\}$

# Our First Technical Building Block: **Descent**

$$\rho_i(w) = \frac{a_i^\top (A^\top \mathrm{diag}(w) A)^{-1} a_i}{w_i^\alpha}$$

The descent step:

$$[\mathrm{Descent}(w, C, \eta)]_i := w_i \left[1 + \eta_i \cdot \frac{\rho_i(w) - 1}{\rho_i(w) + 1}\right].$$

for $i \in C \subseteq \{1, 2, \ldots, m\}$

- Inspired by Newton step

# Our First Technical Building Block: **Descent**

$$\rho_i(w) = \frac{a_i^\top (A^\top \mathrm{diag}(w) A)^{-1} a_i}{w_i^\alpha}$$

The descent step:

$$[\mathrm{Descent}(w, C, \eta)]_i := w_i \left[1 + \eta_i \cdot \frac{\rho_i(w)-1}{\rho_i(w)+1}\right].$$

for $i \in C \subseteq \{1, 2, \ldots, m\}$

- Inspired by Newton step
- Quasi-Newton step using approximate Hessian

# Our First Technical Building Block: **Descent**

$$\rho_i(w) = \frac{a_i^\top (A^\top \mathrm{diag}(w) A)^{-1} a_i}{w_i^\alpha}$$

The descent step:

$$[\mathrm{Descent}(w, C, \eta)]_i := w_i \left[1 + \eta_i \cdot \frac{\rho_i(w) - 1}{\rho_i(w) + 1}\right].$$

for $i \in C \subseteq \{1, 2, \ldots, m\}$

- Inspired by Newton step
- Quasi-Newton step using approximate Hessian
- Key properties:

# Our First Technical Building Block: **Descent**

$$\rho_i(w) = \frac{a_i^\top (A^\top \mathrm{diag}(w) A)^{-1} a_i}{w_i^\alpha}$$

The descent step:

$$[\mathrm{Descent}(w, C, \eta)]_i := w_i \left[1 + \eta_i \cdot \frac{\rho_i(w)-1}{\rho_i(w)+1}\right].$$

for $i \in C \subseteq \{1, 2, \ldots, m\}$

- Inspired by Newton step
- Quasi-Newton step using approximate Hessian
- Key properties:
  - $C = [m]$ and "rounding" condition satisfied: fast progress

# Our First Technical Building Block: **Descent**

$$\rho_i(w) = \frac{a_i^\top (A^\top \mathrm{diag}(w) A)^{-1} a_i}{w_i^\alpha}$$

The descent step:

$$[\mathrm{Descent}(w, C, \eta)]_i := w_i \left[1 + \eta_i \cdot \frac{\rho_i(w)-1}{\rho_i(w)+1}\right].$$

for $i \in C \subseteq \{1, 2, \ldots, m\}$

- Inspired by Newton step
- Quasi-Newton step using approximate Hessian
- Key properties:
  - $C = [m]$ and "rounding" condition satisfied: fast progress
  - Otherwise, fixes the "rounding" condition

# Our Second Technical Building Block: **Rounding**

The rounding condition:

$\sigma_i(w) = w_i \cdot a_i^\top (A^\top WA)^{-1} a_i$

$$\max_{i \in [m]} \rho_i(w) \le 1 + \alpha, \text{ where } \rho_i(w) = \frac{\sigma_i(w)}{w_i^{1+\alpha}}.$$

# Our Second Technical Building Block: **Rounding**

$$\sigma_i(w) = w_i \cdot a_i^\top (A^\top W A)^{-1} a_i$$

The rounding condition:

$$\max_{i\in[m]} \rho_i(w) \le 1+\alpha, \text{ where } \rho_i(w) = \frac{\sigma_i(w)}{w_i^{1+\alpha}}.$$

- Updating $w_i$ updates $\nabla_i \mathcal{F}(w) \approx w_i^\alpha$ for small $\rho_i(w)$.

# Our Second Technical Building Block: **Rounding**

The rounding condition: $\sigma_i(w) = w_i \cdot a_i^\top (A^\top W A)^{-1} a_i$

$$\max_{i \in [m]} \rho_i(w) \leq 1 + \alpha, \text{ where } \rho_i(w) = \frac{\sigma_i(w)}{w_i^{1+\alpha}}.$$

- Updating $w_i$ updates $\nabla_i \mathcal{F}(w) \approx w_i^\alpha$ for small $\rho_i(w)$.
- Critical in upper bounding $\mathcal{F}(w) - \mathcal{F}(\overline{w})$.

# Our Second Technical Building Block: **Rounding**

The rounding condition:

$$\sigma_i(w) = w_i \cdot a_i^\top (A^\top W A)^{-1} a_i$$

$$\max_{i\in[m]} \rho_i(w) \le 1 + \alpha, \text{ where } \rho_i(w) = \frac{\sigma_i(w)}{w_i^{1+\alpha}}.$$

- Updating $w_i$ updates $\nabla_i \mathcal{F}(w) \approx w_i^\alpha$ for small $\rho_i(w)$.
- Critical in upper bounding $\mathcal{F}(w) - \mathcal{F}(\overline{w})$.
  - Critical to turning $\mathrm{poly}(1/\epsilon)$ runtime into $\mathrm{polylog}(1/\epsilon)$.

# Our Second Technical Building Block: **Rounding**

The rounding condition:

$\sigma_i(w) = w_i \cdot a_i^\top (A^\top WA)^{-1} a_i$

$$\max_{i\in[m]} \rho_i(w) \leq 1+\alpha, \text{ where } \rho_i(w) = \frac{\sigma_i(w)}{w_i^{1+\alpha}}.$$

- Updating $w_i$ updates $\nabla_i \mathcal{F}(w) \approx w_i^\alpha$ for small $\rho_i(w)$.
- Critical in upper bounding $\mathcal{F}(w) - \mathcal{F}(\overline{w})$.
  - Critical to turning $\mathrm{poly}(1/\epsilon)$ runtime into $\mathrm{polylog}(1/\epsilon)$.
- Rounding implies a certain ellipsoid being inside a polytope.

# Our Second Technical Building Block: **Rounding**

The rounding condition: $\sigma_i(w) = w_i \cdot a_i^\top (A^\top W A)^{-1} a_i$

$$\max_{i\in[m]} \rho_i(w) \leq 1+\alpha, \text{ where } \rho_i(w) = \frac{\sigma_i(w)}{w_i^{1+\alpha}}.$$

- Updating $w_i$ updates $\nabla_i \mathcal{F}(w) \approx w_i^\alpha$ for small $\rho_i(w)$.
- Critical in upper bounding $\mathcal{F}(w) - \mathcal{F}(\overline{w})$.
  - Critical to turning $\mathrm{poly}(1/\epsilon)$ runtime into $\mathrm{polylog}(1/\epsilon)$.
- Rounding implies a certain ellipsoid being inside a polytope.

Rounding Algorithm:

1. while $C = \{i : \rho_i(w) > 1+\alpha\} \neq \emptyset$,
   - $w \leftarrow \textbf{Descent}(w, C, \frac{1}{\alpha})$
2. Return $w$

# Analysis of Algorithm: **Descent** Step

Want to bound iteration complexity of **Descent** steps, where

$$\mathbf{Descent}(w, C, \eta)]_i := w_i \left[1 + \eta_i \cdot \frac{\rho_i(w)-1}{\rho_i(w)+1}\right]$$

for $i \in C \subseteq \{1, 2, \ldots, m\}$

# Analysis of Algorithm: **Descent** Step

Want to bound iteration complexity of **Descent** steps, where

$$\mathbf{Descent}(w, C, \eta)]_i := w_i \left[1 + \eta_i \cdot \frac{\rho_i(w)-1}{\rho_i(w)+1}\right]$$

for $i \in C \subseteq \{1, 2, \ldots, m\}$

**Lemma 1: Descent Step Convergence**

Each **Descent** step decreases the objective, and the number of **Descent** steps is $O(\alpha^{-1} \log(m/\varepsilon))$.

# Analysis of Algorithm: **Descent** Step

Want to bound iteration complexity of **Descent** steps, where

$$\mathbf{Descent}(w, C, \eta)]_i := w_i \left[1 + \eta_i \cdot \frac{\rho_i(w)-1}{\rho_i(w)+1}\right]$$

for $i \in C \subseteq \{1, 2, \ldots, m\}$

**Lemma 1: Descent Step Convergence**

Each **Descent** step decreases the objective, and the number of **Descent** steps is $O(\alpha^{-1} \log(m/\varepsilon))$.

Proof Idea: Upper bound on sub-optimality, lower bound on progress.

# Analysis of Algorithm: **Descent** Step

**Lemma 2: Descent Step Convergence**

Each **Descent** step decreases the objective, and the number of **Descent** steps is $O(\alpha^{-1}\log(m/\varepsilon))$.

$$\mathcal{F}(w) - \mathcal{F}(\overline{w}) \quad \leq O(\alpha^{-1}) \sum_{i=1}^{m} w_i^{1+\alpha} \frac{(\rho_i(w)-1)^2}{\rho_i(w)+1}$$

# Analysis of Algorithm: **Descent** Step

**Lemma 2: Descent Step Convergence**

Each **Descent** step decreases the objective, and the number of **Descent** steps is $O(\alpha^{-1}\log(m/\varepsilon))$.

$$\mathcal{F}(w) - \mathcal{F}(\overline{w}) \leq O(\alpha^{-1}) \sum_{i=1}^{m} w_i^{1+\alpha} \frac{(\rho_i(w)-1)^2}{\rho_i(w)+1}$$

$$\mathcal{F}(w^+) - \mathcal{F}(w) \leq \sum_{i=1}^{m} -\eta_i w_i^{1+\alpha} \frac{(\rho_i(w)-1)^2}{\rho_i(w)+1}.$$

# Analysis of Algorithm: **Descent** Step

**Lemma 2: Descent Step Convergence**

Each **Descent** step decreases the objective, and the number of **Descent** steps is $O(\alpha^{-1}\log(m/\varepsilon))$.

$$\mathcal{F}(w) - \mathcal{F}(\overline{w}) \le O(\alpha^{-1}) \sum_{i=1}^{m} w_i^{1+\alpha} \frac{(\rho_i(w)-1)^2}{\rho_i(w)+1}$$

$$\mathcal{F}(w^+) - \mathcal{F}(w) \le \sum_{i=1}^{m} -\eta_i w_i^{1+\alpha} \frac{(\rho_i(w)-1)^2}{\rho_i(w)+1}.$$

$$\mathcal{F}(w^+) - \mathcal{F}(\overline{w}) \le \mathcal{F}(w) - \mathcal{F}(\overline{w}) - \frac{1}{\alpha} \sum_{i=1}^{m} w_i^{1+\alpha} \frac{(\rho_i(w)-1)^2}{\rho_i(w)+1}$$

# Analysis of Algorithm: **Descent** Step

**Lemma 2: Descent Step Convergence**

Each **Descent** step decreases the objective, and the number of **Descent** steps is $O(\alpha^{-1}\log(m/\varepsilon))$.

$$\mathcal{F}(w) - \mathcal{F}(\overline{w}) \leq O(\alpha^{-1}) \sum_{i=1}^{m} w_i^{1+\alpha} \frac{(\rho_i(w)-1)^2}{\rho_i(w)+1}$$

$$\mathcal{F}(w^+) - \mathcal{F}(w) \leq \sum_{i=1}^{m} -\eta_i w_i^{1+\alpha} \frac{(\rho_i(w)-1)^2}{\rho_i(w)+1}.$$

$$\mathcal{F}(w^+) - \mathcal{F}(\overline{w}) \leq \mathcal{F}(w) - \mathcal{F}(\overline{w}) - \frac{1}{\alpha} \sum_{i=1}^{m} w_i^{1+\alpha} \frac{(\rho_i(w)-1)^2}{\rho_i(w)+1}$$

$$\leq (1 - O(\alpha))(\mathcal{F}(w) - \mathcal{F}(\overline{w})).$$

# Analysis of Algorithm: **Round** Step

Want to bound iteration complexity *within* one **Round** step.

# Analysis of Algorithm: **Round** Step

Want to bound iteration complexity *within* one **Round** step.

**Lemma 3: Round `while` loop iterations**

The number of iterations inside the `while` loop in one **Round** step is $O(\alpha^{-2}\log(\rho_{\mathsf{max}}(w))$.

# Analysis of Algorithm: **Round** Step

Want to bound iteration complexity *within* one **Round** step.

**Lemma 3: Round `while` loop iterations**

The number of iterations inside the `while` loop in one **Round** step is $O(\alpha^{-2} \log(\rho_{\max}(w))$.

Proof Sketch: Combine **Round**'s termination condition $\rho_{\max}(w) \leq 1 + \alpha$ with the maximum increase in $\rho_{\max}(w)$ per `while` loop .

# Round Step Complexity of `while` Loop: Bounding $\rho_{\max}(w)$

**Lemma 4: Round change in $\rho_{\max}(w)$**

Let $w^+$ be the state of $w$ after one `while` loop of **Round**. Then,

$$\rho_{\max}(w^+) \le \left(1 + \frac{\alpha}{1+\alpha}\right)^{-\alpha} \rho_{\max}(w).$$

$$w_i^+ = w_i + \frac{w_i}{3}\left(\frac{\rho_i(w)-1}{\rho_i(w)+1}\right) \quad \text{//Descent on } i \in C$$

$$\ge w_i + \frac{w_i}{3\alpha}\left(\frac{\alpha}{2+\alpha}\right) \quad //x \to \frac{x-1}{x+1} \text{ is monotone for } x \ge 1$$

Combine with $w_i = w_i^+$ for all $i \notin C$ implies $w^+ \ge w$. Hence:

$$\rho_i(w^+) = [w_i^+]^{-\alpha}\left[A(A^\top \mathrm{diag}(w^+)A)^{-1}A^\top\right]_{ii} \le \left[1 + \frac{\alpha}{3(2+\alpha)}\right]^{-\alpha} \rho_i(w)$$

# Round Step Complexity of `while` Loop: Bounding $\rho_{\max}(w)$

**Lemma 4: Round change in $\rho_{\max}(w)$**

Let $w^+$ be the state of $w$ after one `while` loop of **Round**. Then,

$$\rho_{\max}(w^+) \leq \left(1 + \frac{\alpha}{1+\alpha}\right)^{-\alpha} \rho_{\max}(w).$$

$$w_i^+ = w_i + \frac{w_i}{3}\left(\frac{\rho_i(w) - 1}{\rho_i(w) + 1}\right) \quad \text{//\textbf{Descent} on } i \in C$$

$$\geq w_i + \frac{w_i}{3\alpha}\left(\frac{\alpha}{2+\alpha}\right) \quad //x \to \frac{x-1}{x+1} \text{ is monotone for } x \geq 1$$

Combine with $w_i = w_i^+$ for all $i \notin C$ implies $w^+ \geq w$. Hence:

$$\rho_i(w^+) = [w_i^+]^{-\alpha}\left[A(A^\top \mathrm{diag}(w^+)A)^{-1}A^\top\right]_{ii} \leq \left[1 + \frac{\alpha}{3(2+\alpha)}\right]^{-\alpha} \rho_i(w)$$

# Round Step Complexity of while Loop: Bounding $\rho_{\max}(w)$

**Lemma 4: Round change in $\rho_{\max}(w)$**

Let $w^+$ be the state of $w$ after one while loop of **Round**. Then,

$$\rho_{\max}(w^+) \le \left(1 + \frac{\alpha}{1+\alpha}\right)^{-\alpha} \rho_{\max}(w).$$

$$w_i^+ = w_i + \frac{w_i}{3}\left(\frac{\rho_i(w) - 1}{\rho_i(w) + 1}\right) \quad //\textbf{Descent} \text{ on } i \in C$$

$$\ge w_i + \frac{w_i}{3\alpha}\left(\frac{\alpha}{2+\alpha}\right) \quad //x \to \frac{x-1}{x+1} \text{ is monotone for } x \ge 1$$

Combine with $w_i = w_i^+$ for all $i \notin C$ implies $w^+ \ge w$. Hence:

$$\rho_i(w^+) = [w_i^+]^{-\alpha} \left[A(A^\top \mathrm{diag}(w^+)A)^{-1}A^\top\right]_{ii} \le \left[1 + \frac{\alpha}{3(2+\alpha)}\right]^{-\alpha} \rho_i(w)$$

# Round Step Complexity of while Loop: Bounding $\rho_{\max}(w)$

**Lemma 4: Round change in $\rho_{\max}(w)$**

Let $w^+$ be the state of $w$ after one `while` loop of **Round**. Then,

$$\rho_{\max}(w^+) \le \left(1 + \frac{\alpha}{1+\alpha}\right)^{-\alpha} \rho_{\max}(w).$$

$$w_i^+ = w_i + \frac{w_i}{3}\left(\frac{\rho_i(w) - 1}{\rho_i(w) + 1}\right) \quad // \textbf{Descent on } i \in C$$

$$\ge w_i + \frac{w_i}{3\alpha}\left(\frac{\alpha}{2+\alpha}\right) \quad // x \to \frac{x-1}{x+1} \text{ is monotone for } x \ge 1$$

Combine with $w_i = w_i^+$ for all $i \notin C$ implies $w^+ \ge w$. Hence:

$$\rho_i(w^+) = [w_i^+]^{-\alpha}\left[A(A^\top \mathrm{diag}(w^+)A)^{-1}A^\top\right]_{ii} \le \left[1 + \frac{\alpha}{3(2+\alpha)}\right]^{-\alpha} \rho_i(w)$$

# Conclusion: Total Iteration Complexity

Most basic step: Leverage score computation

# Conclusion: Total Iteration Complexity

Most basic step:

Denote:

Leverage score computation

$\mathcal{R}$ := Steps in Round,
$\mathcal{D}$ := Steps of Descent

# Conclusion: Total Iteration Complexity

Most basic step: Leverage score computation

Denote: $\mathcal{R}$ := Steps in Round, $\mathcal{D}$ := Steps of Descent

Throughout the algorithm, $\mathcal{D} = \mathcal{O}(\alpha^{-1} \log(m/\varepsilon))$

# Conclusion: Total Iteration Complexity

| | |
|---|---|
| Most basic step: | Leverage score computation |
| Denote: | $\mathcal{R}$ := Steps in Round,<br>$\mathcal{D}$ := Steps of Descent |
| Throughout the algorithm, | $\mathcal{D} = \mathcal{O}(\alpha^{-1} \log(m/\varepsilon))$ |
| Between two **Descent** calls: | $\mathcal{R} := \mathcal{O}(\alpha^{-2} \log(\rho_{\max}(w)))$ |

# Conclusion: Total Iteration Complexity

| | |
|---|---|
| Most basic step: | Leverage score computation |
| Denote: | $\mathcal{R}$ := Steps in Round,<br>$\mathcal{D}$ := Steps of Descent |
| Throughout the algorithm, | $\mathcal{D} = \mathcal{O}(\alpha^{-1} \log(m/\varepsilon))$ |
| Between two **Descent** calls: | $\mathcal{R} := \mathcal{O}(\alpha^{-2} \log(\rho_{\max}(w)))$ |
| Initial iteration: | $\mathcal{R} := \mathcal{O}(\alpha^{-2} \log(m/n))$ |

# Conclusion: Total Iteration Complexity

| | |
|---|---|
| Most basic step: | Leverage score computation |
| Denote: | $\mathcal{R}$ := Steps in Round,<br>$\mathcal{D}$ := Steps of Descent |
| Throughout the algorithm, | $\mathcal{D} = \mathcal{O}(\alpha^{-1} \log(m/\varepsilon))$ |
| Between two **Descent** calls: | $\mathcal{R} := \mathcal{O}(\alpha^{-2} \log(\rho_{\max}(w)))$ |
| Initial iteration: | $\mathcal{R} := \mathcal{O}(\alpha^{-2} \log(m/n))$ |
| Between two **Round** calls: | Minimum increase in $w$,<br>Maximum increase in $\rho_{\max}(w)$ |

# Conclusion: Total Iteration Complexity

| | |
|---|---|
| Most basic step: | Leverage score computation |
| Denote: | $\mathcal{R}$ := Steps in Round,<br>$\mathcal{D}$ := Steps of Descent |
| Throughout the algorithm, | $\mathcal{D} = \mathcal{O}(\alpha^{-1} \log(m/\varepsilon))$ |
| Between two **Descent** calls: | $\mathcal{R} := \mathcal{O}(\alpha^{-2} \log(\rho_{\max}(w)))$ |
| Initial iteration: | $\mathcal{R} := \mathcal{O}(\alpha^{-2} \log(m/n))$ |
| Between two **Round** calls: | Minimum increase in $w$,<br>Maximum increase in $\rho_{\max}(w)$ |
| Therefore, over $\mathcal{D}$ iterations: | $\mathcal{R} := \mathcal{O}(\mathcal{D} \cdot \alpha^{-2} \log(m/n))$ |

# Our Contribution

**Lemma 5: Our Main Result (Informal)**

Given a full-rank $A \in \mathbb{R}^{m \times n}$ and $p \geq 4$, we compute $\varepsilon$-approximate $\ell_p$ Lewis weights in $O(p^3 \log(mp/\varepsilon))$ oracle calls.

# Our Contribution

**Lemma 5: Our Main Result (Informal)**

Given a full-rank $A \in \mathbb{R}^{m \times n}$ and $p \geq 4$, we compute $\varepsilon$-approximate $\ell_p$ Lewis weights in $O(p^3 \log(mp/\varepsilon))$ oracle calls.

- Our algorithm has both a parallel and a sequential version.

# Our Contribution

**Lemma 5: Our Main Result (Informal)**

Given a full-rank $A \in \mathbb{R}^{m \times n}$ and $p \geq 4$, we compute $\varepsilon$-approximate $\ell_p$ Lewis weights in $O(p^3 \log(mp/\varepsilon))$ oracle calls.

- Our algorithm has both a parallel and a sequential version.
- Computing gradient of Lee-Sidford barrier to high-precision in polylogarithmic depth.

Thank You!