CS-GY 6763: Lecture 8
Dimension Dependent Optimization

NYU Tandon School of Engineering, Prof. Christopher Musco

ADMINISTRATIVE

- Problem Set 3 is out.
- Quizzes restart this week.
- If you are completing a project, a 1-page proposal is due **end of day today**.
- We are working grading pset 2 and midterms.

We missed a few things due to missed class... Please see last years lecture notes if these topics interest you.

STOCHASTIC METHODS

Main idea: Trade slower convergence (more iterations) for cheaper iterations.

Stochastic Coordinate Descent: Only compute a single random entry of $\nabla f(\mathbf{x})$ on each iteration:

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\mathbf{x}) \\ \frac{\partial f}{\partial x_2}(\mathbf{x}) \\ \vdots \\ \frac{\partial f}{\partial x_d}(\mathbf{x}) \end{bmatrix} \qquad \nabla_i f(\mathbf{x}) = \begin{bmatrix} 0 \\ \frac{\partial f}{\partial x_i}(\mathbf{x}) \\ \vdots \\ 0 \end{bmatrix}$$

Update: $\mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} + \eta \nabla_i f(\mathbf{x}^{(t)})$.

Stochastic Coordinate Descent:

- Choose number of steps T and step size η.
- For $t = 1, \ldots, T$:
  - Pick random $j \in 1, \ldots, d$ uniformly at random.
  - $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \eta \nabla_j f(\mathbf{x}^{(i)})$
- Return $\hat{\mathbf{x}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}^{(t)}$.

Theorem (Stochastic Coordinate Descent convergence)

Given a G-Lipschitz function f with minimizer $\mathbf{x}^*$ and initial point $\mathbf{x}^{(1)}$ with $\|\mathbf{x}^{(1)} - \mathbf{x}^*\|_2 \leq R$, SCD with step size $\eta = \frac{1}{Rd}$ satisfies the guarantee:

$$\mathbb{E}[f(\hat{\mathbf{x}}) - f(\mathbf{x}^*)] \leq \frac{2GR}{\sqrt{T/d}}$$

NON-CONVEX OPTIMIZATION

STATIONARY POINTS

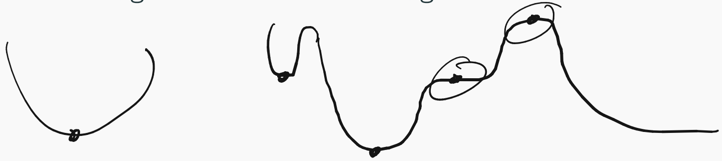
We understand much less about optimizing non-convex functions in comparison to convex functions, but not nothing. In many cases, we're still figuring out the right questions to ask

Definition (Stationary point)

For a differentiable function f, a <u>stationary point</u> is any $\mathbf{x}$ with:

$$\nabla f(\mathbf{x}) = \mathbf{0}$$

local/global minima - local/global maxima - saddle points



STATIONARY POINTS

Reasonable goal: Find an approximate stationary point $\hat{\mathbf{x}}$ with

$$\|\nabla f(\hat{\mathbf{x}})\|_2^2 \leq \epsilon.$$

GRADIENT DESCENT FINDS APPROXIMATE STATIONARY POINTS

Theorem

If GD is run with step size $\eta = \frac{1}{\beta}$ on a differentiable, $\underline{\beta\text{-smooth}}$ function f with global minimum $\mathbf{x}^*$ then after $T = O\left(\frac{\beta[f(\mathbf{x}^{(1)}) - f(\mathbf{x}^*)]}{\epsilon}\right)$ we will find an ϵ-approximate stationary point $\hat{\mathbf{x}}$.

- $\nabla f(\mathbf{x}^{(t)})^T(\mathbf{x}^{(t)} - \mathbf{x}^{(t+1)}) - f(\mathbf{x}^{(t)}) + f(\mathbf{x}^{(t+1)}) \leq \frac{\beta}{2}\|\mathbf{x}^{(t)} - \mathbf{x}^{(t+1)}\|_2^2$.
- $f(\mathbf{x}^{(t+1])}) - f(\mathbf{x}^{(t)}) \leq \frac{\beta}{2}\eta^2\|\nabla f(\mathbf{x}^{(t)})\|_2^2 - \eta\|\nabla f(\mathbf{x}^{(t)})\|_2^2$
- $f(\mathbf{x}^{(t+1])}) - f(\mathbf{x}^{(t)}) \leq \frac{-\eta}{2}\|\nabla f(\mathbf{x}^{(t)})\|_2^2$
- $\frac{1}{T}\sum_{t=1}^{T}\frac{\eta}{2}\|\nabla f(\mathbf{x}^{(t)})\|_2^2 \leq \frac{1}{T}\sum_{t=1}^{T} f(\mathbf{x}^{(t)}) - f(\mathbf{x}^{(t+1)})$
- $\frac{\eta}{2}\min_t \|\nabla f(\mathbf{x}^{(t)})\|_2^2 \leq \frac{1}{T}\left[f(\mathbf{x})^{(1)} - f(\mathbf{x})^{(T)})\right]$

QUESTIONS IN NON-CONVEX OPTIMIZATION

If GD can find a stationary point, are there algorithms which find a stationary point faster using preconditioning, acceleration, stocastic methods, etc.?

QUESTIONS IN NON-CONVEX OPTIMIZATION

What if my function only has global minima and saddle points? Randomized methods (SGD, perturbed gradient methods, etc.) can "escape" stationary points under some minor assumptions.

Example: $\min_{\mathbf{x}} \frac{-\mathbf{x}^T \mathbf{A}^T \mathbf{A} \mathbf{x}}{\mathbf{x}^T \mathbf{x}}$

- **Global minimum**: Top eigenvector of $\mathbf{A}^T\mathbf{A}$ (i.e., top principal component of $\mathbf{A}$).
- **Saddle points:** All other eigenvectors of $\mathbf{A}$.

Useful for lots of other matrix factorization problems beyond vanilla PCA.

DIMENSION DEPENDENT OPTIMIZATION

FIRST ORDER CONVEX OPTIMIZATION

First Order Optimization: Given a convex function f and a convex set $\mathcal{S}$,

Goal: Find $\hat{x} \in \mathcal{S}$ such that $\underline{f(\hat{x})} \leq \underline{\min_{x\in\mathcal{S}} f(x)} + \epsilon$.

Assume we have:

- **Function oracle**: Evaluate $f(x)$ for any x. $1/\epsilon$ $\log(1/\epsilon)$
- **Gradient oracle**: Evaluate $\nabla f(x)$ for any x.
- **Projection oracle**: Evaluate $P_{\mathcal{S}}(x)$ for any x.

Gradient descent requires $O\left(\frac{R^2G^2}{\boxed{\epsilon^2}}\right)$ calls to each oracle to solve the problem.

We were only able to improve the ϵ dependence by making stronger assumptions on f (strong convexity, smoothness).

DIMENSION DEPENDENT BOUND

Alternatively, we can get much better bounds if we are willing to depend on the problem dimension. I.e. on d if $f(\mathbf{x})$ is a function mapping d-dimensional vectors to scalars.

We already know how to do this for a few special functions:

$$\boxed{f(\mathbf{x}) = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2} \quad \text{where} \quad \mathbf{A} \in \mathbb{R}^{n \times d}.$$

$$\nabla f(x) = \cancel{2}(A^T A x - A^T b) = 0 \qquad A^T A x^* = A^T b$$

$$x^* = \underbrace{(A^T A)^{-1} A^T b}$$

$$O(nd^2) + O(d^3)$$

DIMENSION DEPENDENT BOUND

$f(x) = 0$ for some $x \in S$

Let $f(\mathbf{x})$ be bounded between $[-B, B]$ on $\underline{\mathcal{S}}$.

Theorem (Dimension Dependent Convex Optimization)

There is an algorithm (the Center-of-Gravity Method) which finds $\hat{\mathbf{x}}$ satisfying $f(\hat{\mathbf{x}}) \leq \min_{\mathbf{x}\in\mathcal{S}} f(\mathbf{x}) + \underline{\epsilon}$ using $O\underline{(d \log(B/\epsilon))}$ calls to a function and gradient oracle for f.

Caveat: Assumes we have some representation of $\mathcal{S}$, not just a projection oracle. We will discuss this more later.

Note: For starting radius R and $\max_{\mathbf{x}} \|\nabla f(\mathbf{x})\|_2 = G$, without loss of generality we have $\underline{B = O(RG)}$.

CENTER OF GRAVITY METHOD

$s_1, \ldots, s_n$

A few basic ingredients:

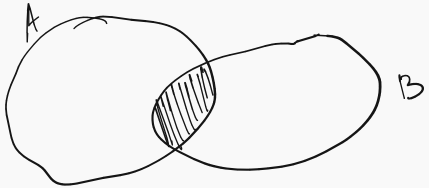
1. The center-of-gravity of a convex set $\mathcal{S}$ is defined as:

centroid

$$c = \frac{\int_{x\in\mathcal{S}} x\,dx}{\text{vol}(\mathcal{S})} = \frac{\int_{x\in\mathcal{S}} x\,dx}{\int_{x\in\mathcal{S}} 1\,dx}$$

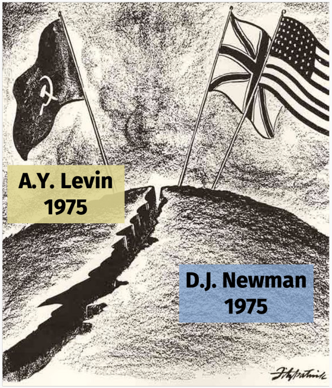
$$\frac{1}{n}\sum_{i=1}^{n} s_i$$

2. For two convex sets $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A}\cap\mathcal{B}$ is convex. Proof by picture:



CENTER OF GRAVITY METHOD

Natural "cutting plane" method. Developed simultaneous on opposite sides of iron curtain.



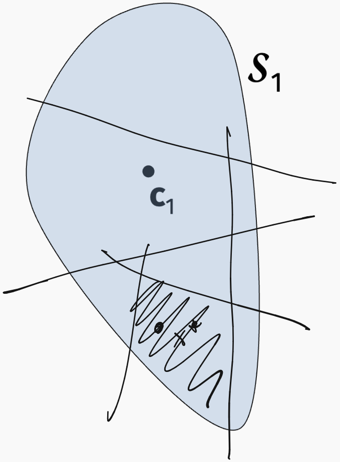
Not used in practice (we will discuss why) but the basic idea underlies many algorithms that are.

CENTER OF GRAVITY METHOD

min $f(x)$ $x \in S = S_1$

Natural “cutting plane” method.

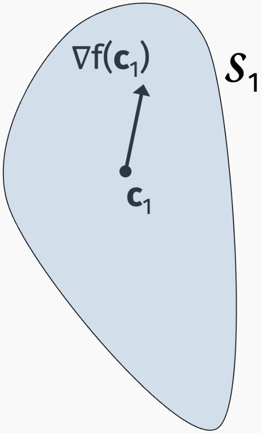
- $\underline{\mathcal{S}_1 = \mathcal{S}}$
- For $t = 1, \ldots, T$:
    - $\mathbf{c}_t$ = center of gravity of $\mathcal{S}_t$.
    - Compute $\nabla f(\mathbf{c}_t)$.
    - $\mathcal{H} = \{\mathbf{x} \,|\, \langle \nabla f(\mathbf{c}_t), \mathbf{x} - \mathbf{c}_t \rangle \leq 0\}$.
    - $\mathcal{S}_{t+1} = \mathcal{S}_t \cap H$
- Return $\hat{\mathbf{x}} = \arg\min_t f(\mathbf{c}_t)$



CENTER OF GRAVITY METHOD

Natural "cutting plane" method.

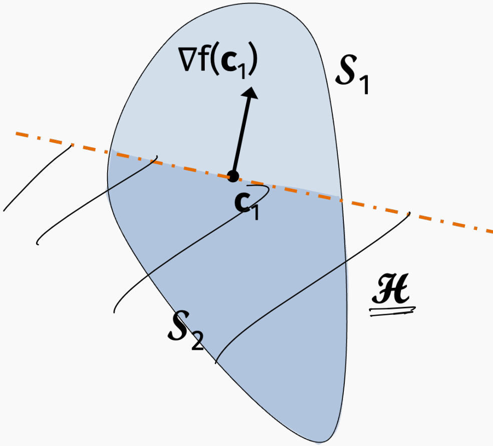
- $\mathcal{S}_1 = \mathcal{S}$
- For $t = 1, \ldots, T$:
  - $\mathbf{c}_t$ = center of gravity of $\mathcal{S}_t$.
  - Compute $\nabla f(\mathbf{c}_t)$.
  - $\mathcal{H} = \{\mathbf{x} \,|\, \langle \nabla f(\mathbf{c}_t), \mathbf{x} - \mathbf{c}_t \rangle \leq 0\}$.
  - $\mathcal{S}_{t+1} = \mathcal{S}_t \cap H$
- Return $\hat{\mathbf{x}} = \arg\min_t f(\mathbf{c}_t)$



CENTER OF GRAVITY METHOD

Natural "cutting plane" method.

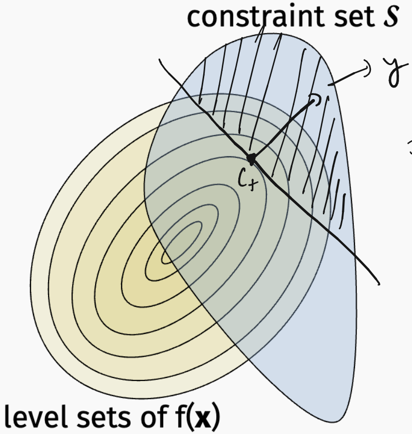
- $\mathcal{S}_1 = \mathcal{S}$
- For $t = 1, \ldots, T$:
    - $\underline{\mathbf{c}_t} =$ center of gravity of $\mathcal{S}_t$.
    - Compute $\nabla f(\mathbf{c}_t)$.
    - $\mathcal{H} = \{\mathbf{x} | \langle \nabla \underline{f(\mathbf{c}_t)}, \mathbf{x} - \mathbf{c}_t \rangle \overset{=}{\leq} 0\}$.
    - $\underline{\mathcal{S}_{t+1}} = \underline{\mathcal{S}_t} \cap H$
- Return $\hat{\mathbf{x}} = \arg\min_t f(\mathbf{c}_t)$



CENTER OF GRAVITY METHOD

Intuitively, why does it make sense to search in $\mathcal{S}_t \cap \mathcal{H}$ where:

$$\mathcal{H} = \{\mathbf{x} \big| \langle \nabla f(\mathbf{c}_t), \mathbf{x} - \mathbf{c}_t \rangle \leq 0\}?$$

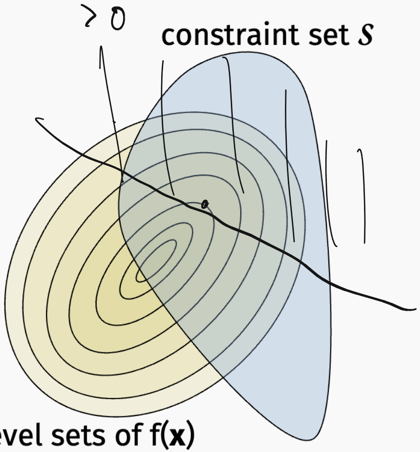


$f(y) > f(c_t)$

CENTER OF GRAVITY METHOD

Intuitively, why does it make sense to search in $\mathcal{S}_t \cap \mathcal{H}$ where:

$$\boxed{\mathcal{H}} = \{\mathbf{x} \,|\, \langle \nabla f(\mathbf{c}_t), \mathbf{x} - \mathbf{c}_t \rangle \leq 0\}?$$



By convexity, if $\mathbf{y} \notin \{\mathcal{S}_t \cap \mathcal{H}\}$ then:

positive

$$\begin{aligned} \underline{f(\mathbf{y})} &\geq \underline{f(\mathbf{c}_t)} + \boxed{\underline{\langle \nabla f(\mathbf{c}_t), \mathbf{y} - \mathbf{c}_t \rangle}} \\ &> f(\mathbf{c}_t) \end{aligned}$$

CONVERGENCE THEOREM

Theorem (Center-of-Gravity Convergence)

Let f be a convex function with values in $[-B, B]$. Let $\hat{\mathbf{x}}$ be the output of the center-of-gravity method run for T iterations. Then:

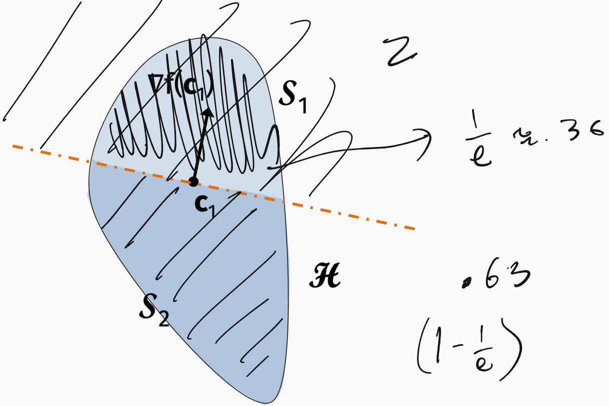
$$\underline{f}(\hat{\mathbf{x}}) - f(\underline{\mathbf{x}^*}) \leq \underline{2B \left(1 - \overbrace{\frac{1}{e}}^{.63}\right)^{T/d}} \leq 2Be^{-T/3d}.$$

If we set $T = 3d \log(2B/\epsilon)$, then $\underline{f(\hat{\mathbf{x}}) - f(\mathbf{x}^*) \leq \epsilon}$.

$$\hat{x} = \underset{c_t}{\operatorname{argmin}} f(c_t)$$

KEY GEOMETRIC TOOL

Want to argue that, at every step of the algorithm, we "cut off" a large portion of the convex set we are searching over:

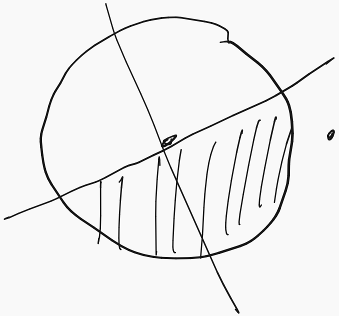


KEY GEOMETRIC TOOL

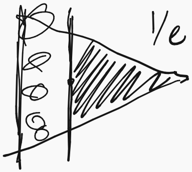
Theorem (Grünbaum's Theorem)

For any convex set $\mathcal{S}$ with center-of-gravity $\mathbf{c}$*, and any halfspace* $\mathcal{Z} = \{\mathbf{x} \,|\, \langle \mathbf{a}, \mathbf{x} - \mathbf{c}\rangle \leq 0\}$ *then:*

$$\frac{\mathrm{vol}(\mathcal{S} \cap \mathcal{Z})}{\mathrm{vol}(\mathcal{S})} \geq \frac{1}{e} \approx .368$$



$.368 = \frac{1}{e}$



KEY GEOMETRIC TOOL

Want to argue that, at every step of the algorithm, we "cut off" a large portion of the convex set we are searching over.

Theorem (Grünbaum's Theorem)

For any convex set $\mathcal{S}$ *with center-of-gravity* $\mathbf{c}$*, and any halfspace* $\mathcal{Z} = \{\mathbf{x} \,|\, \langle \mathbf{a}, \mathbf{x} - \mathbf{c} \rangle \leq 0\}$ *then:*

$$\frac{\text{vol}(\mathcal{S} \cap \mathcal{Z})}{\text{vol}(\mathcal{S})} \geq \frac{1}{e} \approx .368$$

Let $\mathcal{Z}$ be the compliment of $\mathcal{H}$ from the algorithm. Then we cut off at least a $1/e$ fraction of the convex body on every iteration.

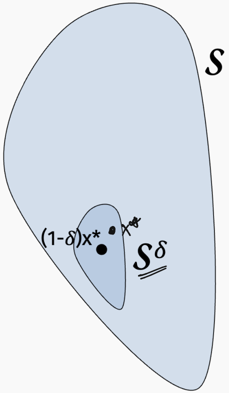
Corollary: After t steps, $\underline{\text{vol}(\mathcal{S}_t)} \leq \left(1 - \frac{1}{e}\right)^t \underline{\text{vol}(\mathcal{S})}$.

CONVERGENCE PROOF

Let δ be a small parameter to be chosen later.

Let $\mathcal{S}^\delta = \{\underline{(1-\delta)}\mathbf{x}^* + \underline{\delta}\mathbf{x} \mid \text{for } \mathbf{x} \in \mathcal{S}\}$. $0 < \delta < 1$

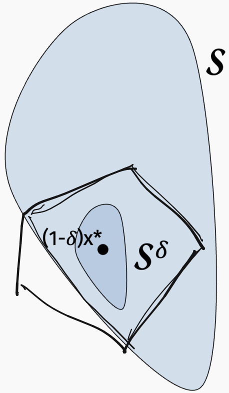
$x : x^{*}$



Claim: Every point $\mathbf{y}$ in $\mathcal{S}^\delta$ has good function value.

CONVERGENCE PROOF

$y = (1-\delta) x^* + \delta x$



For any $\mathbf{y} \in \mathcal{S}^\delta$:

$$\begin{aligned} \underline{f(\mathbf{y})} &= \underline{f((1-\delta)\mathbf{x}^* + \delta\mathbf{x})} \\ &\leq \underline{(1-\delta)f(\mathbf{x}^*)} + \underline{\delta f(\mathbf{x})} \\ &\leq \underline{f(\mathbf{x}^*) - \delta f(\mathbf{x}^*)} + \delta f(\mathbf{x}) \\ &\leq \underline{f(\mathbf{x}^*) + 2B\delta}. \end{aligned}$$

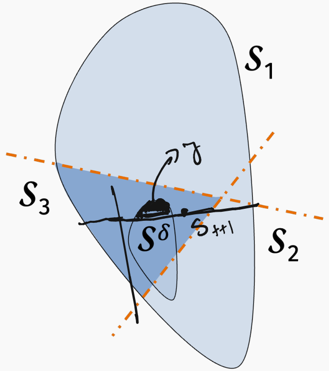
bounded between [-B, B]

CONVERGENCE PROOF

$$\left(\left(1-\frac{1}{e}\right)^{t/d}\right)^{d} = \left(1-\frac{1}{e}\right)^{t}$$

We also have: $\underline{\text{vol}(\mathcal{S}^\delta)} \leq \underline{\delta^d}\, \underline{\text{vol}(\mathcal{S})}$.

Set $\delta = \left(1-\frac{1}{e}\right)^{t/d}$. After $t+1$ steps, $\underline{\text{vol}(\mathcal{S}_t)} \leqslant \underline{\text{vol}(\mathcal{S}^\delta)}$. $\text{vol}(S_t) \leq \left(1-\frac{1}{e}\right)^{t} \text{vol}(S)$



$$\underline{\langle \nabla f(c_i), y - c_i\rangle > 0}$$

We must have "chopped off" at least one point $\mathbf{y}$ in $\mathcal{S}^\delta$ by the time we $\underline{\text{reach step } t+1}$.

Claim: For some centroid $\overset{c_i \text{ in}}{\mathbf{c}_1, \ldots, \mathbf{c}_t}$,

$$\boxed{\underline{2B\delta}} \geq f(\mathbf{y}) \geq \underline{f(\mathbf{c}_i) + \langle \nabla f(\mathbf{c}_i), \mathbf{y} - \mathbf{c}_t\rangle}$$
$$> \underline{f(\mathbf{c}_i)}.$$

Algorithm returns $\underline{\arg\min_{\mathbf{c}_i} f(\mathbf{c}_i)}$.

CONVERGENCE THEOREM

Theorem (Center-of-Gravity Convergence)

Let f be a convex function with values in $[-B, B]$. Let $\hat{\mathbf{x}}$ be the output of the center-of-gravity method run for T iterations. Then:

$$f(\hat{\mathbf{x}}) - f(\mathbf{x}^*) \leq 2B\left(1 - \frac{1}{e}\right)^{T/d} \leq 2Be^{-T/3d}.$$

If we set $T = O\left(d \log(\bigcirc\!\!\!\!B/\epsilon)\right)$, then $f(\hat{\mathbf{x}}) - f(\mathbf{x}^*) \leq \epsilon$.

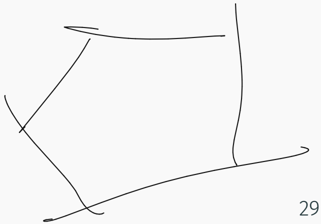
In terms of <u>gradient-oracle</u> complexity, this is essentially optimal. So why isn't the algorithm used?

CENTROID COMPUTATION

In general computing the centroid is hard. #P-hard even when when $\mathcal{S}$ is an intersection of half-spaces (a polytope).

Even if the problem isn't hard for your starting convex body $\mathcal{S}$, it likely will become hard for $\mathcal{S} \cap \mathcal{H}_1 \cap \mathcal{H}_2 \cap \mathcal{H}_3 \ldots$.

So while the oracle complexity of dimension-dependent optimization was settled, in the 70s a number of basic questions in terms of computational complexity.



LINEAR PROGRAMMING

Linear programs (LPs) are one of the most basic convex constrained, convex optimization problems:

Let $\mathbf{c} \in \mathbb{R}^d, \mathbf{b} \in \mathbb{R}^n, \mathbf{A} \in \mathbb{R}^{n \times d}$ be fixed vectors that define the problem, and let $\mathbf{x}$ be our variable parameter.

$$\min \underline{f(\mathbf{x})} = \underline{\mathbf{c}^T \mathbf{x}}$$
$$\text{subject to } \underline{\mathbf{A}\mathbf{x}} \geq \underline{\mathbf{b}}.$$

Think about $\underline{\mathbf{A}\mathbf{x} \geq \mathbf{b}}$ as a union of half-space constraints:

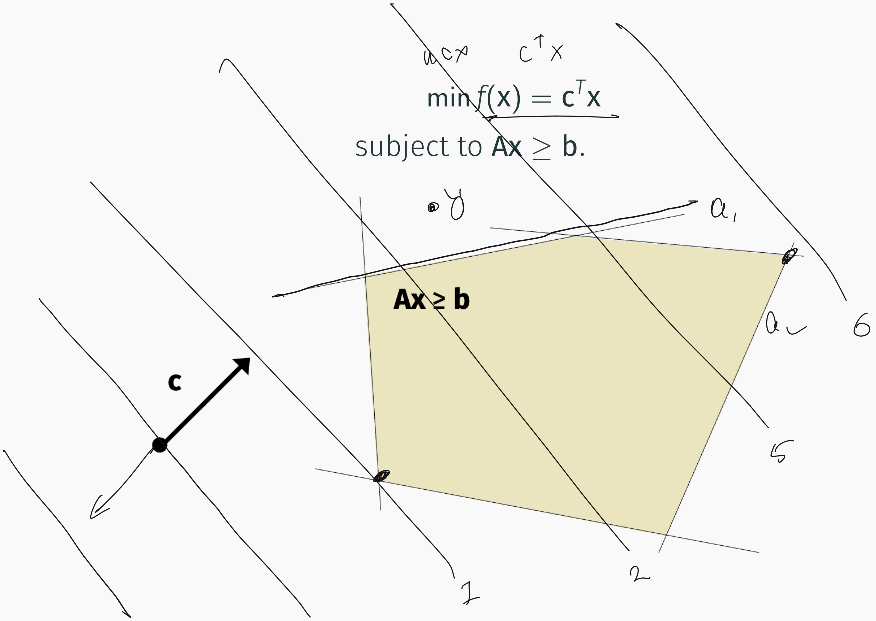
$$\{\mathbf{x} : \mathbf{a}_1^T \mathbf{x} \geq b_1\}$$
$$\{\mathbf{x} : \underline{\mathbf{a}_2^T \mathbf{x}} \geq \underline{b_2}\}$$
$$\vdots$$
$$\{\mathbf{x} : \mathbf{a}_n^T \mathbf{x} \geq b_n\}$$

LINEAR PROGRAMMING

max $c^T x$

$$\min f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$$

subject to $\mathbf{A}\mathbf{x} \geq \mathbf{b}$.



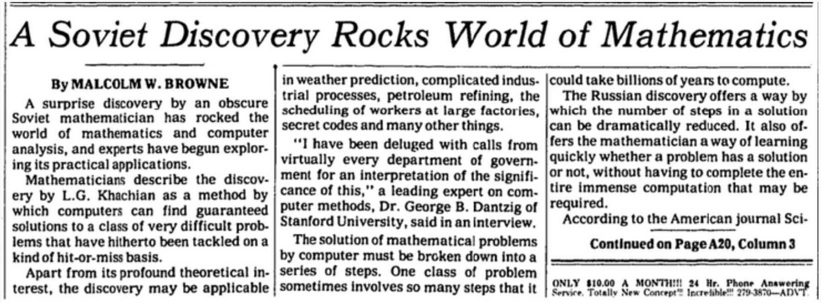
LINEAR PROGRAMMING APPLICATIONS

- Classic optimization applications: industrial resource optimization problems were killer app in the 70s.
- Robust regression: $\min_x \|Ax - b\|_1$.
- *L*1 constrained regression: $\min_x \|x\|_1$ subject to $Ax = b$. Lots of applications in sparse recovery/compressed sensing.
- Solve $\min_x \|Ax - b\|_\infty$.
- Polynomial time algorithms for Markov Decision Processes.
- Many combinatorial optimization problems can be solved via LP relaxations.

LINEAR PROGRAMMING

Theorem (Khachiyan, 1979)
Assume $n = d$. The ellipsoid method solves any linear program with L-bit integer valued constraints in $O(n^4 L)$ time. I.e. linear programming is in (weakly) polynomial time!

Using a relatively simple center-of-gravity like method!



A Soviet Discovery Rocks World of Mathematics

By MALCOLM W. BROWNE

A surprise discovery by an obscure Soviet mathematician has rocked the world of mathematics and computer analysis, and experts have begun exploring its practical applications.

Mathematicians describe the discovery by L.G. Khachian as a method by which computers can find guaranteed solutions to a class of very difficult problems that have hitherto been tackled on a kind of hit-or-miss basis.

Apart from its profound theoretical interest, the discovery may be applicable in weather prediction, complicated industrial processes, petroleum refining, the scheduling of workers at large factories, secret codes and many other things.

"I have been deluged with calls from virtually every department of government for an interpretation of the significance of this," a leading expert on computer methods, Dr. George B. Dantzig of Stanford University, said in an interview.

The solution of mathematical problems by computer must be broken down into a series of steps. One class of problem sometimes involves so many steps that it could take billions of years to compute.

The Russian discovery offers a way by which the number of steps in a solution can be dramatically reduced. It also offers the mathematician a way of learning quickly whether a problem has a solution or not, without having to complete the entire immense computation that may be required.

According to the American journal Sci-

Continued on Page A20, Column 3

ONLY $10.00 A MONTH!!! 24 Hr. Phone Answering Service. Totally New Concept!! Incredible!!! 279-3870—ADVT.

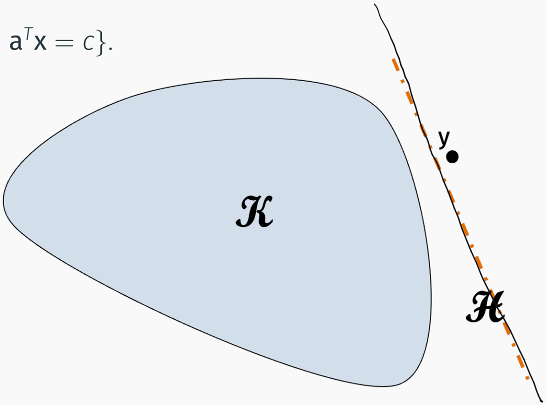
Front page of New York Times, November 9, 1979.

PROBLEM SIMPLIFICATION

Simplifying the problem: Given a convex set $\mathcal{K}$ via access to separation oracle $S_{\mathcal{K}}$ for the set, determine if $\mathcal{K}$ is empty, or otherwise return any point $\mathbf{x} \in \mathcal{K}$.

$$S_k(\mathbf{y}) = \begin{cases} \emptyset & \text{if } \mathbf{y} \in \mathcal{K}. \\ \text{seperating hyperplane } (\mathbf{a}, c) & \text{if } \mathbf{y} \notin \mathcal{K}. \end{cases}$$

Let $\mathcal{H} = \{\mathbf{x} : \mathbf{a}^T \mathbf{x} = c\}$.



SEPARATION ORACLE

Example: How would you implement a seperation oracle for a polytope $\{x : Ax \geq b\}$.

Ay

$a_1^T y \geq b_1$

$\underline{a_2^T y < b_2}$ $\quad a = a_2 \quad c = b_2$

$\vdots$

$a_n^T y \geq b_n$

FROM MEMBERSHIP TO OPTIMIZATION

$\{x : f(x) \leq c\} \cup S$

Original problem:

$$\underline{\min_{\mathbf{x}} f(\mathbf{x})} \text{ subject to } \underline{\mathbf{x} \in \mathcal{S}}$$

How can we reduce to (determining if a convex set $\mathcal{K}$ is empty or not?)

S

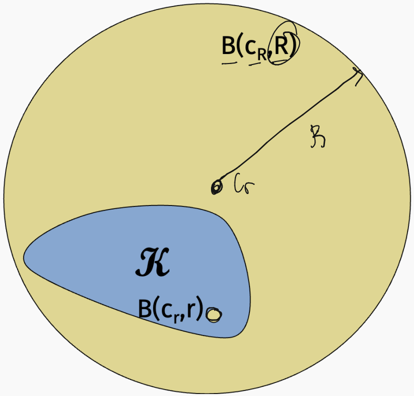
Binary search! For a convex function $f(\mathbf{x})$, $\underline{\{\mathbf{x} : f(\mathbf{x}) \leq c\}}$ is convex, and you can get a seperation oracle via the gradient.

- Start with upper bound and lower bounds u and l on optimal solution (can be obtained for many problems).
- Check if the convex set $\mathcal{S} \cap \{\mathbf{x} : f(\mathbf{x}) \leq (\underline{u + l})/2\}$ contains a point.
- Update $u = (u + l)/2$ if it does, $l = (u + l)/2$ if not.
- Continue until $|u - l| \leq \epsilon$.

ELLIPSOID METHOD SKETCH

Goal of ellipsoid algorithm: Solve "Is $\mathcal{K}$ empty or not?" given a seperation oracle for $\mathcal{K}$ under the assumptions that:

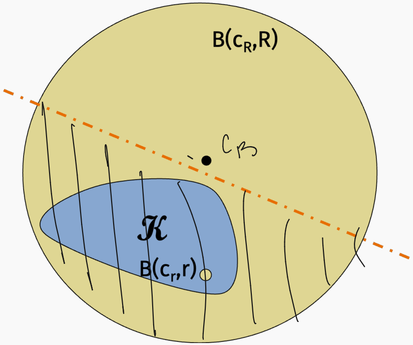
1. $\mathcal{K} \subset B(\mathbf{c}_R, R)$.
2. If non-empty, $\mathcal{K}$ contains $B(\mathbf{c}_r, r)$ for some $r < R$.



ELLIPSOID METHOD SKETCH

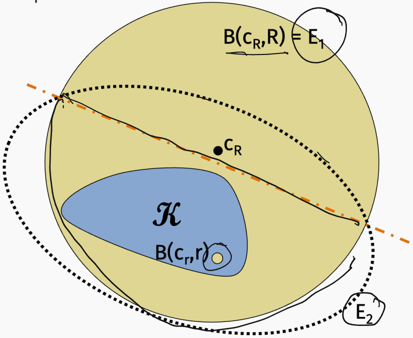
Iterative method similar to center-of-gravity:

1. Check if center $\mathbf{c}_R$ of $\underline{B(\mathbf{c}_R, R)}$ is in $\mathcal{K}$.
2. If it is, we are done.
3. If not, cut search space in half, using seperating hyperplane.



ELLIPSOID METHOD SKETCH

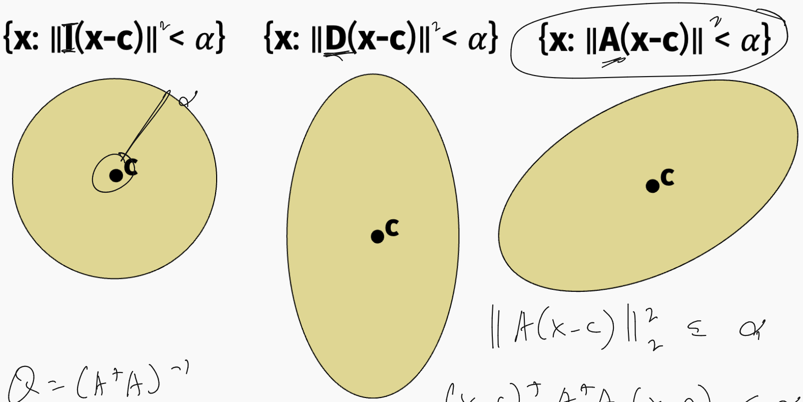
Key insight: Before moving on, approximate new search region by something that we can easily compute the centroid of. Specifically an ellipse!!



Produce a sequence of ellipses that always contain $\mathcal{K}$ and decrease in volume: $B(\mathbf{c}_R, R) = E_1, E_2, \ldots$. Once we get to an ellipse with volume $\leq B(\mathbf{c}_r, r)$, we know that $\mathcal{K}$ must be empty.

ELLIPSE

An ellipse is a convex set of the form: $\{\mathbf{x} : \|\underline{\mathbf{A}(\mathbf{x}-\mathbf{c})}\|_2^2 \leq \alpha\}$ for some constant c and matrix $\mathbf{A}$. The center-of-mass is $\mathbf{c}$.



$\{x: \|\underline{I}(x-c)\|^2 < \alpha\}$
$\{x: \|\underline{D}(x-c)\|^2 < \alpha\}$
$\{x: \|\underline{A}(x-c)\|^2 < \alpha\}$
α
c
c
c

$\|A(x-c)\|_2^2 \leq \alpha$

$(x-c)^T \underline{A^T A} (x-c) \leq \alpha$

$Q = (A^T A)^{-1}$

Often re-parameterized to say that the ellipse is all $\mathbf{x}$ with

$\underline{\{\mathbf{x} : (\mathbf{x}-\mathbf{c})^T \underline{\mathbf{Q}}^{-1}(\mathbf{x}-\mathbf{c}) \leq 1\}}$

ELLIPSOID UPDATE

There is a closed form solution for the equation of the smallest ellipse containing a given half-ellipse. I.e. let $\mathbf{E}_i$ have parameters $\mathbf{Q}_i, \mathbf{c}_i$ and consider the half-ellipse:

$$\underline{\mathbf{E}_i} \cap \{\underline{\mathbf{x} : \mathbf{a}_i^T \mathbf{x} \leq \mathbf{a}_i^T \mathbf{c}_i}\}.$$

Then $\underline{\mathbf{E}_{i+1}}$ is the ellipse with parameters:

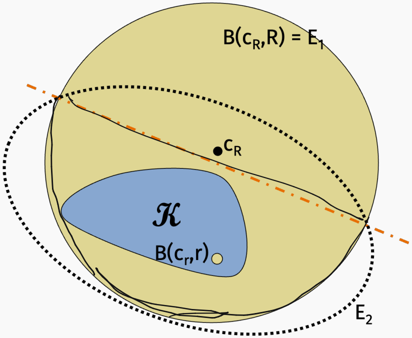
$$\mathbf{Q}_{i+1} = \frac{d^2}{d^2-1}\left(\mathbf{Q}_i - \frac{2}{d+1}\mathbf{h}\mathbf{h}^T\right) \qquad \mathbf{c}_{i+1} = \mathbf{c}_i - \frac{1}{n+1}\mathbf{h},$$

where $\mathbf{h} = \sqrt{\mathbf{a}_i^T \mathbf{Q}_i \mathbf{a}_i} \cdot \mathbf{a}_i$.

GEOMETRIC OBSERVATION

Claim: $\text{vol}(\underline{E_{i+1}}) \leq (\underline{1 - \frac{1}{2d}})\,\text{vol}(E_i)$. $\left(1 - \frac{1}{2d}\right)^{2d} \approx \frac{1}{e}$

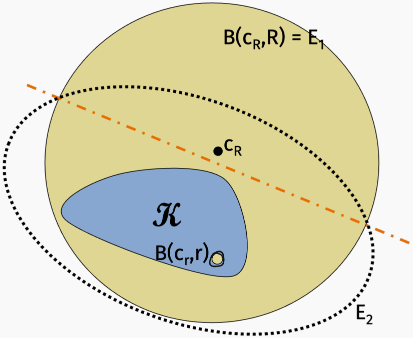
Proof: Via reduction to the "isotropic case". I will post a proof on the course website if you are interested.



Not as good as the $(1 - \frac{1}{e})$ constant-factor volume reduction we got from center-of-gravity, but still very good!

GEOMETRIC OBSERVATION

Claim: $\text{vol}(E_{i+1}) \leq \underline{(1 - \frac{1}{2d})} \text{vol}(E_i)$



After $O(d)$ iterations, we reduce the volume by a constant.

In total require $O(d^2 \log(R/r))$ iterations to solve the problem.

ellipsoid for LPs

Theorem (Khachiyan, 1979)

Assume $n = d$. The ellipsoid method solves any linear program with L-bit integer valued constraints in $O(n^4 L)$ time. I.e. linear programming is in (weakly) polynomial time!

The method works for any convex program.

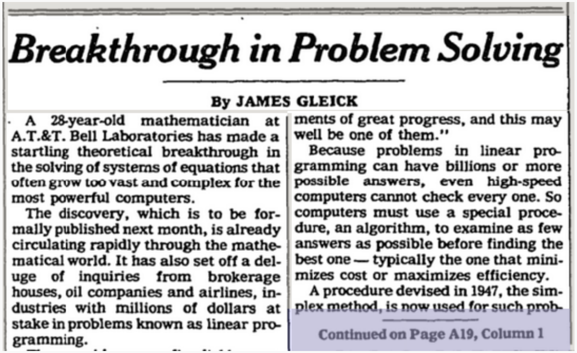
For LPs, we have an $O(nd)$ time seperation oracle, and ellipsoid update take $O(d^2)$ time.

Careful analysis of the binary search method, how to set B_r and B_R, etc. leads to the final runtime bound.

INTERIOR POINT METHODS

Theorem (Karmarkar, ~~1979~~ 1984)

Assume $n = d$. The <u>*interior point method*</u> *solves any linear program with L-bit integer valued constraints in $O(n^{3.5}L)$ time.*



Breakthrough in Problem Solving

By JAMES GLEICK

A 28-year-old mathematician at A.T.&T. Bell Laboratories has made a startling theoretical breakthrough in the solving of systems of equations that often grow too vast and complex for the most powerful computers.

The discovery, which is to be formally published next month, is already circulating rapidly through the mathematical world. It has also set off a deluge of inquiries from brokerage houses, oil companies and airlines, industries with millions of dollars at stake in problems known as linear programming.

ments of great progress, and this may well be one of them."

Because problems in linear programming can have billions or more possible answers, even high-speed computers cannot check every one. So computers must use a special procedure, an algorithm, to examine as few answers as possible before finding the best one — typically the one that minimizes cost or maximizes efficiency.

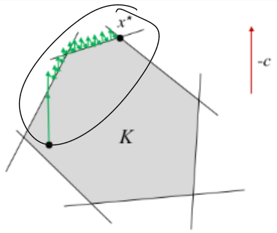
A procedure devised in 1947, the simplex method, is now used for such prob-

Continued on Page A19, Column 1

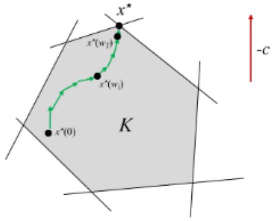
Front page of New York Times, November 19, 1984.

INTERIOR POINT METHODS

Will post some notes on the website.



(a) Gradient Descent optimization path.



(b) Ideal Interior Point optimization path.

POLYNOMIAL TIME LINEAR PROGRAMMING

Both results had a huge impact on the theory of optimization, although at the time neither the ellipsoid method or interior point method were faster than a heuristic known at the Simplex Method.

These days, improved interior point methods compete with and often outperform simplex.