Mathematics of Data: From Theory to Computation

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 13: Disciplined convex optimization

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-556 (Fall 2015)

License Information for Mathematics of Data Slides

Recommended readings

- A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*, 2001.
- A. Nemirovski, *Introduction to linear optimization,* 2012.
- F. Alizadeh and D. Goldfarb, "Second-order cone programming," *Math. Program., Ser. B*, 2003.
- L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM Rev.*, 1996.
- A. Nemirovski, *Interior point polynomial time methods in convex programming*, 2004.

Motivation

Example (Convex Problem)

$$
\begin{array}{ll}
\min\limits_{\mathbf{x}\in\mathbb{R}^p} & F(\mathbf{x}) \\
\text{s.t.} & g_i(\mathbf{x}) \leq 0 \qquad , i = 1, \ldots, s \\
& \mathbf{A}_j\mathbf{x} - \mathbf{b}_j = 0, \ j = 1, \ldots, t \\
& \mathbf{x} \in \mathcal{X}
\end{array}
$$

- $\mathcal{X}$ is a set such that the set of solutions is a nonempty set
- $g_i(\mathbf{x})$ are convex for $i = 0, \ldots, s$

Motivation

Example (Convex Problem)

$$
\begin{array}{ll}
\min\limits_{\mathbf{x}\in\mathbb{R}^p} & F(\mathbf{x}) \\
\text{s.t.} & g_i(\mathbf{x}) \le 0 \qquad , \; i = 1, \ldots, s \\
 & \mathbf{A}_j\mathbf{x} - \mathbf{b}_j = 0, \; j = 1, \ldots, t \\
 & \mathbf{x} \in \mathcal{X}
\end{array}
$$

- $\mathcal{X}$ is a set such that the set of solutions is a nonempty set
- $g_i(\mathbf{x})$ are convex for $i = 0, \ldots, s$

Approach 1 - Previous lectures

Design special purpose software

- Increased convergence speed
- Non-reusable
- Hard to design
- Solid background needed

Motivation

Example (Convex Problem)

$$
\begin{array}{ll}
\min\limits_{\mathbf{x}\in\mathbb{R}^p} & F(\mathbf{x}) \\
\text{s.t.} & g_i(\mathbf{x}) \leq 0 \qquad , \ i = 1, \dots, s \\
& \mathbf{A}_j\mathbf{x} - \mathbf{b}_j = 0, \ j = 1, \dots, t \\
& \mathbf{x} \in \mathcal{X}
\end{array}
$$

- $\mathcal{X}$ is a set such that the set of solutions is a nonempty set
- $g_i(\mathbf{x})$ are convex for $i = 0, \dots, s$

Approach 1 - Previous lectures

Design special purpose software

- Increased convergence speed
- Non-reusable
- Hard to design
- Solid background needed

Approach 2 - This lecture

Structured convex forms

- Less efficient per particular instance
- Readily available software
- Optimized solvers
- Minimal expertise required

Good news

"One size fits all"

$$\text{LP} \subset \text{QP} \subset \text{QCQP} \subset \text{SOCP} \subset \text{SDP}$$

Good news: we need one solver!

- Today - Disciplined Convex Programming (DCP)
  1. DCP
     - Linear programming (LP)
     - Quadratic programming (QP)
     - Quadratically constrained quadratic programming (QCQP)
     - Second order conic programming (SOCP)
     - Semidefinite programming (SDP)
  2. Methods
     - Simplex method
     - Interior point methods

Linear Programming (LP)

A linear program (LP) is the problem of minimizing a linear function subject to finitely many linear equality and inequality constraints.

Definition (LP in the canonical form)

An LP in the canonical form is given by

$$\mathrm{opt} = \min_{\mathbf{x}} \left\{ \boldsymbol{c}^T \mathbf{x} : \mathbf{x} \in \mathbb{R}^p, \mathbf{A}\mathbf{x} \leq \mathbf{b} \right\},$$

for some $\boldsymbol{c} \in \mathbb{R}^p$, $\mathbf{A} \in \mathbb{R}^{n \times p}$, and $\mathbf{b} \in \mathbb{R}^n$.

- Any LP can be converted to an equivalent one in the canonical form.
- A linear equality constraint $\mathbf{B}\mathbf{x} = \mathbf{d}$ is equivalent to two linear inequality constraints $\mathbf{B}\mathbf{x} \leq \mathbf{d}$ and $-\mathbf{B}\mathbf{x} \leq \mathbf{d}$, and can be written as

$$\begin{bmatrix} \mathbf{B} \\ -\mathbf{B} \end{bmatrix} \mathbf{x} \leq \begin{bmatrix} \mathbf{d} \\ \mathbf{d} \end{bmatrix}.$$

Application 1: Basis pursuit

Example (Basis pursuit [4])

Recall the Gaussian linear model $\mathbf{b} = \mathbf{A}\mathbf{x}^\natural + \mathbf{w}$, and assume that $\mathbf{x}^\natural \in \mathbb{R}^p$ is sparse. The basis pursuit estimator for $\mathbf{x}^\natural$ is given by

$$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}} \left\{ \|\mathbf{x}\|_1 : \mathbf{x} \in \mathbb{R}^p, \mathbf{A}\mathbf{x} = \mathbf{b} \right\},$$

for some $\mathbf{A} \in \mathbb{R}^{n \times p}$ and $\mathbf{b} \in \mathbb{R}^p$.

We have used methods for constrained minimization in Lectures 11 and 12 to solve it.

LP formulation

The optimization problem is equivalent to

$$\min_{\mathbf{x}_+, \mathbf{x}_-} \left\{ \mathbf{1}^T (\mathbf{x}_+ - \mathbf{x}_-) : \mathbf{x}_+, \mathbf{x}_- \geq 0, \mathbf{A}(\mathbf{x}_+ - \mathbf{x}_-) = \mathbf{b} \right\},$$

which is an LP, where $\mathbf{1} := (1, 1, \ldots, 1) \in \mathbb{R}^p$ [4]. Another equivalent LP formulation is given by [11]

$$\min_{\mathbf{x}, \mathbf{u}} \left\{ \mathbf{1}^T \mathbf{u} : \mathbf{u} \geq 0, -\mathbf{u} \leq \mathbf{x} \leq \mathbf{u}, \mathbf{A}\mathbf{x} = \mathbf{b} \right\},$$

where $\mathbf{u}$ denotes the "contour" of $\mathbf{x}$.

Application 2: Dantzig selector

Example (Dantzig selector [3])

Recall the Gaussian linear model $\mathbf{b} = \mathbf{A}\mathbf{x}^\natural + \mathbf{w}$, and assume that $\mathbf{x}^\natural \in \mathbb{R}^p$ is sparse. It is shown in [2] that the Dantzig selector defined as

$$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}} \left\{ \|\mathbf{x}\|_1 : \mathbf{x} \in \mathbb{R}^p, \left\| \mathbf{A}^T(\mathbf{b} - \mathbf{A}\mathbf{x}) \right\|_\infty \leq \lambda \right\},$$

for some properly chosen $\lambda > 0$ behaves similarly to the Lasso, and hence can be used to estimate $\mathbf{x}^\natural$.

LP formulation

The optimization problem is equivalent to

$$\min_{\mathbf{x},\mathbf{u}} \left\{ \mathbf{1}^T\mathbf{u} : \mathbf{u} \geq 0, -\mathbf{u} \leq \mathbf{x} \leq \mathbf{u}, -\lambda\mathbf{1} \leq \mathbf{A}^T(\mathbf{b} - \mathbf{A}\mathbf{x}) \leq \lambda\mathbf{1} \right\},$$

which is an LP, where we used the “contour” trick as in the previous slide.

Application 3: Maximum flow

Example (Maximum flow)

Let $G = (\mathcal{V}, \mathcal{E})$ be a directed graph, where $\mathcal{V}$ denotes the set of nodes, and $\mathcal{E} \subseteq \{(u, v) : u, v \in \mathcal{V}\}$ denotes the directed edges. Let $s, t \in \mathcal{E}$. The maximum flow problem seeks to find the *flow* $f_{u,v}$ for all $(u, v) \in \mathcal{V}$ that maximizes the sum flow from s to t subject to

- **capacity constraint:** $f_{u,v} \leq c_{u,v}$ for all $(u, v) \in \mathcal{E}$, where $c_{u,v}$ are given capacity constraints, and
- **flow conservation:** $\sum_{u:(u,v)\in\mathcal{E}} f_{u,v} = \sum_{w:(v,w)\in\mathcal{E}} f_{v,w}$ for all $v \in \mathcal{V} \setminus \{s, t\}$.

LP formulation

The maximum flow problem is equivalent to

$$\max_{x_{u,v}} \left\{ \sum_{v:(s,v)\in\mathcal{E}} x_{s,v} : x_{u,v} \geq 0 \text{ for all } (u, v) \in \mathcal{E}, \right.$$

$$\left. \text{capacity constraint \& flow conservation} \right\}.$$

Note that this is an LP.

The dual problem of an LP

Recall an LP in the canonical form is given by

$$\text{opt} = \min_{\mathbf{x}} \left\{ \boldsymbol{c}^T \mathbf{x} : \mathbf{x} \in \mathbb{R}^p, \mathbf{A}\mathbf{x} \leq \mathbf{b} \right\},$$

for some $\boldsymbol{c} \in \mathbb{R}^p$, $\mathbf{A} \in \mathbb{R}^{n \times p}$, and $\mathbf{b} \in \mathbb{R}^n$.

Definition (The dual problem)

The corresponding dual problem is given by

$$\text{opt}^* = \min_{\boldsymbol{\lambda}} \left\{ \mathbf{b}^T \boldsymbol{\lambda} : \boldsymbol{\lambda} \in \mathbb{R}^n, \boldsymbol{\lambda} \geq 0, \mathbf{A}^T \boldsymbol{\lambda} = -\boldsymbol{c} \right\}.$$

Intuition

The primal problem is equivalent to maximizing $-\boldsymbol{c}^T \mathbf{x}$. Let $\boldsymbol{\lambda} \in \mathbb{R}^n$ satisfying $\boldsymbol{\lambda} \geq 0$ and $\mathbf{A}^T \boldsymbol{\lambda} = -\boldsymbol{c}$. Then

$$-\boldsymbol{c}^T \mathbf{x} = (\mathbf{A}^T \boldsymbol{\lambda})^T \mathbf{x} = \boldsymbol{\lambda}^T (\mathbf{A}\mathbf{x}) \leq \boldsymbol{\lambda}^T \mathbf{b}.$$

Therefore, the dual problem minimizes an upper bound of the original (primal) problem.

LP duality theorem

Theorem (Weak and strong LP duality)

Consider an LP and the corresponding dual problem. Then

- ***Symmetry:*** *The dual problem of the dual problem is equivalent to the primal problem.*
- ***Weak duality:*** *For any pair of feasible points* $(\mathbf{x}, \boldsymbol{\lambda})$*, we have*

$$G(\mathbf{x}, \boldsymbol{\lambda}) := \mathbf{b}^T \boldsymbol{\lambda} - (-\boldsymbol{c})^T \mathbf{x} \geq 0,$$

  where G *is called the duality gap.*
- ***Strong duality:*** *If the primal problem has a finite optimal value, so does the dual problem, and* $\mathrm{opt}^* = \mathrm{opt}$*.*

Application of weak duality

- If the optimal objective value of the primal problem is $-\infty$, then the dual problem is not feasible.
- If the optimal objective value of the dual problem is $-\infty$, then the primal problem is not feasible.

* Application of strong duality: the max-flow min-cut theorem

Let $G = (\mathcal{V}, \mathcal{E})$ be a directed graph, where $\mathcal{V}$ denotes the set of nodes, and $\mathcal{E} \subseteq \{(u, v) : u, v \in \mathcal{V}\}$ denotes the directed edges. Let $c_{u,v}$ be given capacity constraints for each $(u, v) \in \mathcal{E}$. Let $s, t \in \mathcal{E}$.

Example (Minimum cut)

The minimum cut problem seeks to find a partition $\mathcal{S}, \mathcal{T}$ of $\mathcal{V}$ that minimizes the cut capacity $\sum_{(u,v)\in\mathcal{E}:u\in\mathcal{S},v\in\mathcal{T}} c_{u,v}$, subject to $s \in \mathcal{S}$, and $t \in \mathcal{T}$.

- The minimum cut capacity poses a bottleneck of the maximum flow from s to t.

Theorem (Max-flow min-cut theorem [5])

The maximum sum flow from s to t equals the minimum cut capacity between s and t.

Sketch of the proof [18].

The minimum cut problem is the dual of the maximum flow problem. Apply strong duality. $\square$

Geometry of an LP

Definition (Extreme point)

A point $\mathbf{x}$ in a convex set $\mathcal{X}$ is an extreme point, if there does not exist $\alpha \in (0,1)$ such that $\mathbf{x} = \alpha\mathbf{u} + (1-\alpha)\mathbf{v}$ for some $\mathbf{u}, \mathbf{v} \in \mathcal{X}$.

Theorem (Krein-Milman theorem [8])

A non-empty bounded convex set is the convex hull of the set of all its extreme points.

Proposition

If the feasible set of an LP does not contain a line, then one of the extreme points of the feasible set is a minimizer.

Proof.

By the Krein-Milman theorem, any point $\mathbf{x}$ in the feasible set can be written as $\mathbf{x} = \sum_{i=1}^{m} \alpha_i \mathbf{v}_i, \alpha_i \geq 0, \sum_{i=1}^{m} \alpha_i = 1$, where $\mathbf{v}_1, \ldots, \mathbf{v}_m$ denote the extreme points.

Then for any linear objective function $f(\mathbf{x}) := \boldsymbol{c}^T\mathbf{x}$ for some vector $\boldsymbol{c}$, we have $f(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i f(\mathbf{v}_i) \leq \max_i f(\mathbf{v}_i)$. □

Geometry of an LP contd.

Definition (Polyhedron)

A non-empty set $\mathcal{X} \subseteq \mathbb{R}^p$ is polyhedral, or a polyhedron, if

$$\mathcal{X} = \{\mathbf{x} : \mathbf{x} \in \mathbb{R}^p, \mathbf{A}\mathbf{x} \leq \mathbf{b}\},$$

for some $\mathbf{A} \in \mathbb{R}^{n \times p}$ and $\mathbf{b} \in \mathbb{R}^n$.

- The feasible set of an LP of the canonical form is polyhedral.

Proposition ([11])

A point $\mathbf{x}$ in a polyhedron $\mathcal{X} := \{\mathbf{x} : \mathbf{x} \in \mathbb{R}^p, \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$ is an extreme point, if and only if it is the unique solution of $\mathbf{A}_{\mathcal{I}}\mathbf{v} = \mathbf{b}_{\mathcal{I}}$ for some $\mathcal{I} \subseteq \{1, \ldots, p\}$.

Corollary

For any polyhedron, the number of extreme points is finite.

Proof.

The number of systems of linear equations of the form $\mathbf{A}_{\mathcal{I}}\mathbf{v} = \mathbf{b}_{\mathcal{I}}$ is finite. □

- Hence we only need to compare the function values on a finite number of points.

Simplex methods

Consider an LP of the canonical form. Assume its feasible set does not contain a line.

Definition (Face and improving edge)

A face of the feasible set is a subset of the feasible set, for which there exists a non-empty $\mathcal{I} \subseteq \{1, \ldots, n\}$ such that all of its elements satisfy $\mathbf{A}_{\mathcal{I}}\mathbf{x} = \mathbf{b}_{\mathcal{I}}$.

An improving edge is a one-dimensional face of the feasible set, along which the objective value decreases.

Prototype of simplex methods

A typical simplex method

1. $\mathbf{v} \leftarrow$ an extreme point of the feasible set
2. **While** there is an improving edge $\mathbf{e}$ involving $\mathbf{v}$

$$\mathbf{v} \leftarrow \text{the othe end of } \mathbf{e}$$

3. **Output** $\mathbf{v}$.

- The rule of finding an improving edge in Step 2 is called a pivot rule.
- The complexity of simplex methods depends on the design of the pivot rule, which determines the number of iterations.

Complexity of simplex methods

A long-standing open problem

- Analyses imply the number of iterations for simplex methods cannot be polynomial in the worst case [1].
- Empirical performance (on non-pathological cases) yields $\mathcal{O}(n)$.

Is there a pivot rule for the simplex algorithm that yields a polynomial number of iterations? (See Problem 9 of Smale's Mathematical Problems for the Next Century [15])

Partial answers

- The smallest number of iterations can be upper-bounded by $\mathcal{O}(p^{\log n})$ [6].
- Simplex methods can have polynomial expected number of iterations for random $\mathbf{A}$, $\mathbf{b}$, and c, while these results are not practical [16].
- **Smoothed analysis:** For any LP of the canonical form, for which $\mathbf{A}$ and $\mathbf{b}$ are perturbed by a small random noise, a simplex method has polynomial expected number of iterations [16].

Quadratically constrained quadratic programming (QCQP)

Definition (Quadratic program (QP))

A QP is an optimization problem of the form

$$\mathrm{opt} = \min_{\mathbf{x}} \left\{ \mathbf{x}^T \mathbf{P} \mathbf{x} + 2\mathbf{q}^T \mathbf{x} + r : \mathbf{x} \in \mathbb{R}^p, \mathbf{A}\mathbf{x} \leq \mathbf{b} \right\},$$

for some $\mathbf{A} \in \mathbb{R}^{n \times p}$, $\mathbf{b} \in \mathbb{R}^n$, $\mathbf{P} \in \mathbb{R}^{p \times p}$, $\mathbf{q} \in \mathbf{R}^p$, and $r \in \mathbb{R}$.

- A QP is a convex optimization problem if $\mathbf{P} \succeq 0$.

Definition (Quadratically Constrained Quadratic Program (QCQP))

A QCQP is an optimization problem of the form

$$\begin{aligned} \mathrm{opt} = \min_{\mathbf{x}} \Big\{ & \mathbf{x}^T \mathbf{P}_0 \mathbf{x} + 2\mathbf{q}_0^T \mathbf{x} + r_0 : \\ & \mathbf{x} \in \mathbb{R}^p, \mathbf{x}^T \mathbf{P}_i \mathbf{x} + 2\mathbf{q}_i^T \mathbf{x} + r_i \leq 0 \text{ for all } i = 1, \ldots, m \Big\}, \end{aligned}$$

for some $\mathbf{P}_i \in \mathbb{R}^{p \times p}$, $\mathbf{q}_i \in \mathbb{R}^p$, and $r_i \in \mathbb{R}$.

- A QCQP is a convex optimization problem if $\mathbf{P}_i \succeq 0$ for all i.

Application 1: Portfolio optimization

Example (Markowitz portfolio optimization (Nobel Prize) [9])

Given a collection of n possible investments with return rates $r_1, \ldots, r_n$, modeled as RVs with mean $\mathbb{E}[r_i] = \mu_i$ and variance $\sigma_i = \mathbb{E}[(r_i - \mu_i)^2]$, the goal is to maximize the return of a portfolio represented by ratio of available capital invested x_i in each of them. The return of the portofolio is $R = \sum_{i=1}^{n} x_i r_i$ and $\mathbb{E}[R] = \mathbf{x}^T \mu$, $\mathbb{E}[(R - \mathbb{E}[R])^2] = \mathbf{x}^T \mathbf{G} \mathbf{x}$, where $G_{i,j} = \rho_{i,j} \sigma_i \sigma_j$ and $\rho_{i,j}$ is the corelation between investment return i and j.

The convex optimization formulation of this problem is:

$$
\begin{aligned}
\min_{\mathbf{x} \in \mathbb{R}^n} \quad & \mathbf{x}^T \mu - \kappa \mathbf{x}^T \mathbf{G} \mathbf{x} \\
\text{s.t.} \quad & \sum_{i=1}^{n} x_i = 1 \\
& \mathbf{x} \geq 0 \,,
\end{aligned}
$$

where κ is a parameter for the "risk".

Application 2: Sequential Quadratic Programming

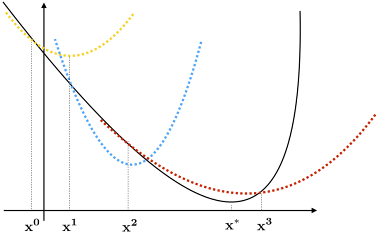
Definition (Sequential Quadratic Programming)

To solve a given convex program

$$\min_{\mathbf{x}\in\mathcal{D}} F(\mathbf{x})$$
$$\text{s.t. } g_i(\mathbf{x}) \le 0,\ i = 1,\ldots,s$$
$$h_j(\mathbf{x}) = 0,\ j = 1,\ldots,t$$

we solve a series of QPs

$$\min_{\mathbf{x}\in\mathcal{D}} F(\mathbf{x}) + \nabla F(\mathbf{x}^k)(\mathbf{x}-\mathbf{x}^k) + \frac{1}{2}(\mathbf{x}-\mathbf{x}^k)^T\nabla^2 F(\mathbf{x}^k)(\mathbf{x}-\mathbf{x}^k)$$
$$\text{s.t. } g_i(\mathbf{x}) + \nabla g_i(\mathbf{x}^k)(\mathbf{x}-\mathbf{x}^k) \le 0,\ i = 1,\ldots,s$$
$$h_j(\mathbf{x}) + \nabla h_j(\mathbf{x}^k)(\mathbf{x}-\mathbf{x}^k) = 0,\ j = 1,\ldots,t$$



Second-order cone programming (SOCP)

Definition (Second-order cone (Lorentz cone))

A second-order cone is a set of the form $\mathcal{L} = \left\{(\mathbf{x}, t) : \mathbf{x} \in \mathbb{R}^p, \|\mathbf{x}\|_2 \leq t\right\} \subseteq \mathbb{R}^{p+1}$.

Definition (Partial ordering induced by a second-order cone)

Let $\mathcal{L}$ be a second-order cone. The partial ordering induced by $\mathcal{L}$ is defined as

$$\mathbf{x} \preceq_{\mathcal{L}} \mathbf{y} \text{ if and only if } \mathbf{y} - \mathbf{x} \in \mathcal{L}^{p+1}.$$

- Especially, $(\mathbf{x}, t) \preceq_{\mathcal{L}} \mathbf{0}$ if and only if $\|\mathbf{x}\|_2 \leq t$.

Definition (Second-order cone program (SOCP))

An SOCP is an optimization problem of the form

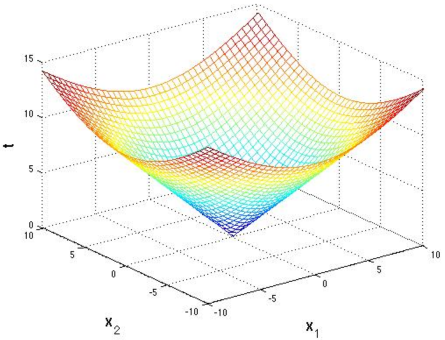
$$\text{opt} = \min_{\mathbf{x}_1, \ldots, \mathbf{x}_m} \left\{ \sum_{i=1}^{m} \boldsymbol{c}_i^T \mathbf{x}_i : \mathbf{x}_i \in \mathbb{R}^p, \sum_{i=1}^{m} \mathbf{A}_i \mathbf{x}_i = \mathbf{b}, \mathbf{x}_i \preceq_{\mathcal{L}} \mathbf{0} \right\},$$

for some $\mathbf{A}_i \in \mathbb{R}^{n \times p}$, $\mathbf{b} \in \mathbb{R}^n$, and $\boldsymbol{c}_i \in \mathbb{R}^p$.

Illustration of a second-order cone

Definition (Second-order cone (Lorentz cone))

A second-order cone is a set of the form $\mathcal{L}^{p+1} = \left\{(\mathbf{x}, t) : \mathbf{x} \in \mathbb{R}^p, \|\mathbf{x}\|_2 \leq t\right\}$.



Application: Basis pursuit denoising

Example (Basis pursuit denoising)

Recall that the basis pursuit denoising estimator (Lecture ?) is given by

$$\hat{\mathbf{x}} \in \arg\min_{\mathbf{x}} \left\{ \|\mathbf{x}\|_1 : \mathbf{x} \in \mathbb{R}^p, \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2 \leq \sigma \right\},$$

for some $\mathbf{A} \in \mathbb{R}^{n \times p}$, $\mathbf{b} \in \mathbb{R}^n$, and $\sigma > 0$.

We could use methods from Lectures 11 and 12 to solve it.

SOCP formulation

The optimization problem is equivalent to

$$\min_{\mathbf{x}_+, \mathbf{x}_-, \mathbf{y}, z} \Big\{ \mathbf{1}^T (\mathbf{x}_+ + \mathbf{x}_-) : \mathbf{y} = \mathbf{b} - \mathbf{A}(\mathbf{x}_+ - \mathbf{x}_-), z = \sigma,$$

$$(z, \mathbf{y}) \preceq_{\mathcal{L}} \mathbf{0}, ((\mathbf{x}_+)_i, 0) \preceq_{\mathcal{L}} \mathbf{0}, ((\mathbf{x}_-)_i, 0) \preceq_{\mathcal{L}} \mathbf{0} \text{ for all } i \Big\},$$

where $(\mathbf{x}_+)_i$ and $(\mathbf{x}_-)_i$ denote the i-th element of $\mathbf{x}_+$ and $\mathbf{x}_-$, respectively.

Semidefinite programming (SDP)

Definition (Semidefinite program (SDP))

A semidefinite program is an optimization problem of the form

$$\mathrm{opt} = \min_{\mathbf{x}} \left\{ \boldsymbol{c}^T \mathbf{x} : \mathbf{x} \in \mathbb{R}^p, \mathbf{F}_0 + \sum_{i=1}^{p} \mathbf{x}_i \mathbf{F}_i \succeq \mathbf{0} \right\},$$

for some $\boldsymbol{c} \in \mathbb{R}^p$ and symmetric matrices $\mathbf{F}_0, \ldots, \mathbf{F}_p \in \mathbb{R}^{m \times m}$.

Reminder

- The **eigenvalue decomposition** of a square matrix, $\mathbf{A} \in \mathbb{R}^{n \times n}$, is given by:

$$\mathbf{A} = \mathbf{X}\boldsymbol{\Lambda}\mathbf{X}^{-1}$$

- $\mathbf{A} \succeq 0$ if all its eigenvalues are **nonnegative** i.e. $\lambda_{\min}(\mathbf{A}) \geq 0$.
- Similarly, $\mathbf{A} \succ 0$ if all its eigenvalues are **nonnegative** i.e. $\lambda_{\min}(\mathbf{A}) > 0$.

Examples: LP and maximum eigenvalue minimization

Example (LP as SDP)

The LP in the canonical form

$$\mathrm{opt} = \min_{\mathbf{x}} \left\{ \boldsymbol{c}^T \mathbf{x} : \mathbf{x} \in \mathbb{R}^p, \mathbf{A}\mathbf{x} \leq \mathbf{b} \right\}$$

is equivalent to the SDP

$$\mathrm{opt} = \min_{\mathbf{x}} \left\{ \boldsymbol{c}^T \mathbf{x} : \mathbf{x} \in \mathbb{R}^p, \mathrm{diag}(\mathbf{b} - \mathbf{A}\mathbf{x}) \succeq \mathbf{0} \right\}.$$

Example (Maximum eigenvalue minimization [17])

Define $\mathbf{A}(\mathbf{x}) := \mathbf{A}_0 + \sum_{i=1}^p \mathbf{x}_i \mathbf{A}_i$ for symmetric matrices $\mathbf{A}_0, \ldots, \mathbf{A}_p$. The problem of minimizing the maximum eigenvalue of $\mathbf{A}(\mathbf{x})$,

$$\mathrm{opt} = \min_{\mathbf{x}} \{ \lambda_{\max}(\mathbf{A}(\mathbf{x})) : \mathbf{x} \in \mathbb{R}^p \},$$

is equivalent to the SDP

$$\mathrm{opt} = \min_{t, \mathbf{x}} \{ t : t \in \mathbb{R}, \mathbf{x} \in \mathbb{R}^p, t\mathbf{I} - \mathbf{A}(\mathbf{x}) \succeq \mathbf{0} \}.$$

Schur complement

Definition (Schur complement)

Consider a symmetric matrix $\mathbf{X} \in \mathbb{R}^{n\times n}$ given by

$$\mathbf{X} = \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{pmatrix},$$

for some symmetric matrix $\mathbf{A} \in \mathbb{R}^{m\times m}$.

If $\mathbf{A}$ is non-singular, then

$$\mathbf{S} := \mathbf{C} - \mathbf{B}^T\mathbf{A}^{-1}\mathbf{B}$$

is called the Schur complement of $\mathbf{A}$ in $\mathbf{X}$.

Useful properties:

- $\det(\mathbf{X}) = \det(\mathbf{A})\det(\mathbf{S})$
- $\mathbf{X} \succ \mathbf{0} \Leftrightarrow \mathbf{A} \succ \mathbf{0}$ and $\mathbf{S} \succ \mathbf{0}$
- If $\mathbf{A} \succ \mathbf{0}$, then $\mathbf{X} \succeq \mathbf{0} \Leftrightarrow \mathbf{S} \succeq \mathbf{0}$

Example:

- $(\mathbf{A}\mathbf{x}+\mathbf{b})^T(\mathbf{A}\mathbf{x}+\mathbf{b}) - \mathbf{c}^T\mathbf{x} - d \leq 0 \Leftrightarrow \begin{pmatrix} \mathbf{I} & \mathbf{A}\mathbf{x}+\mathbf{b} \\ (\mathbf{A}\mathbf{x}+\mathbf{b})^T & \mathbf{c}^T\mathbf{x}+d \end{pmatrix} \succeq \mathbf{0}$

QCQP as SDP

Definition (Quadratically Constrained Quadratic Program (QCQP))

A QCQP is an optimization problem of the form

$$\mathrm{opt} = \min_{\mathbf{x}} \Big\{ \mathbf{x}^T \mathbf{P}_0 \mathbf{x} + 2\mathbf{q}_0^T \mathbf{x} + r_0 : \\ \mathbf{x} \in \mathbb{R}^p, \mathbf{x}^T \mathbf{P}_i \mathbf{x} + 2\mathbf{q}_i^T \mathbf{x} + r_i \leq 0 \text{ for all } i = 1, \ldots, m \Big\},$$

for some $\mathbf{P}_i \in \mathbb{R}^{p \times p}$, $\mathbf{q}_i \in \mathbb{R}^p$, and $r_i \in \mathbb{R}$.

SDP formulation

Assume that $\mathbf{P}_i \succeq \mathbf{0}$, and hence can be decomposed as $\mathbf{P}_i = \mathbf{M}_i^T \mathbf{M}_i$ for all i. The QCQP is equivalent to the SDP given by

$$\mathrm{opt} = \min_{\mathbf{x}, t} \left\{ t : \mathbf{x} \in \mathbb{R}^p, t \in \mathbb{R}, \begin{bmatrix} \mathbf{I} & \mathbf{M}_0 \mathbf{x} \\ (\mathbf{M}_0 \mathbf{x})^T & -2\mathbf{q}^T \mathbf{x} - r_0 + t \end{bmatrix} \succeq \mathbf{0}, \\ \begin{bmatrix} \mathbf{I} & \mathbf{M}_i \mathbf{x} \\ (\mathbf{M}_i \mathbf{x})^T & -2\mathbf{q}^T \mathbf{x} - r_i \end{bmatrix} \succeq \mathbf{0} \right\}.$$

SOCP as SDP

Definition (Second-order cone program (SOCP))

An SOCP is an optimization problem of the form

$$\mathrm{opt} = \min_{\mathbf{x}_1,\ldots,\mathbf{x}_m} \left\{ \sum_{i=1}^{m} \boldsymbol{c}_i^T \mathbf{x}_i : \mathbf{x}_i \in \mathbb{R}^p, \sum_{i=1}^{m} \mathbf{A}_i \mathbf{x}_i = \mathbf{b}, \mathbf{x}_i \preceq_{\mathcal{L}} \mathbf{0} \right\},$$

for some $\mathbf{A}_i \in \mathbb{R}^{n \times p}$, $\mathbf{b} \in \mathbb{R}^n$, and $\boldsymbol{c}_i \in \mathbb{R}^p$.

SDP formulation

Write each $\mathbf{x}_i$ as $\mathbf{x}_i = (\mathbf{v}_i, t_i)^T$ for all i. The constraint $\mathbf{x}_i \preceq_{\mathcal{L}} \mathbf{0}$ is equivalent to

$$\begin{bmatrix} t_i \mathbf{I} & \mathbf{v}_i \\ \mathbf{v}_i^T & t_i \end{bmatrix} \succeq \mathbf{0},$$

for all i.

Applications: Matrix completion

Example (Matrix completion)

Let $\mathbf{X}^\natural \in \mathbb{R}^{p\times p}$ be an unknown low-rank matrix, and we want to estimate $\mathbf{X}^\natural$ given a linear operator $\mathcal{A}:\mathbb{R}^{p\times p}\to\mathbb{R}^n$ and $\mathbf{b} := \mathcal{A}\left(\mathbf{X}^\natural\right) \in \mathbb{R}^n$. An estimator is given by

$$\hat{\mathbf{X}} \in \arg\min_{\mathbf{X}} \left\{ \|\mathbf{X}\|_* : \mathbf{X} \in \mathbb{R}^{p\times p}, \mathcal{A}(\mathbf{X}) = \mathbf{b} \right\}.$$

We could use methods from Lectures 11 and 12 to solve it.

SDP formulation [13]

The optimization problem is equivalent to the SDP given by

$$\min_{\mathbf{X},\mathbf{Y},\mathbf{Z}} \left\{ \frac{1}{2}\left[\mathrm{Tr}(\mathbf{Y}) + \mathrm{Tr}(\mathbf{Z})\right] : \mathbf{X},\mathbf{Y},\mathbf{Z} \in \mathbb{R}^{p\times p}, \mathcal{A}(\mathbf{X}) = \mathbf{b}, \begin{bmatrix} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{bmatrix} \succeq \mathbf{0} \right\}.$$

- The proof in [13] uses the duality of SDP. We show another proof in the next slide.

Applications: Matrix completion contd.

Proposition

For any matrix $\mathbf{X} \in \mathbb{R}^{p \times p}$, we have

$$\|\mathbf{X}\|_* = \min_{\mathbf{Y},\mathbf{Z}} \left\{ \frac{1}{2} \left[\mathrm{Tr}(\mathbf{Y}) + \mathrm{Tr}(\mathbf{Z}) \right] : \mathbf{Y}, \mathbf{Z} \in \mathbb{R}^{p \times p}, \left[\begin{array}{cc} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{array} \right] \succeq \mathbf{0} \right\}.$$

Proof.

Consider the SVD of $\mathbf{X}$, $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}$. If

$$\left[\begin{array}{cc} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{array} \right] \succeq \mathbf{0},$$

then we have

$$\left[\begin{array}{cc} \mathbf{U}^T & -\mathbf{V}^T \end{array} \right] \left[\begin{array}{cc} \mathbf{Y} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Z} \end{array} \right] \left[\begin{array}{c} \mathbf{U} \\ \mathbf{V} \end{array} \right] = \mathbf{U}^T\mathbf{Y}\mathbf{U} + \mathbf{V}^T\mathbf{Z}\mathbf{V} - 2\mathbf{\Sigma} \succeq \mathbf{0},$$

and hence

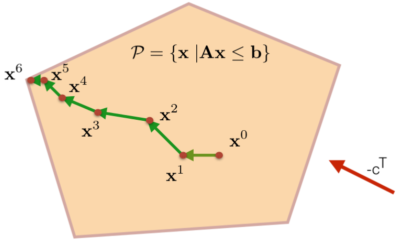
$$\mathrm{Tr}(\mathbf{U}^T\mathbf{Y}\mathbf{U} + \mathbf{V}^T\mathbf{Z}\mathbf{V} - 2\mathbf{\Sigma}) = \mathrm{Tr}(\mathbf{Y}) + \mathrm{Tr}(\mathbf{Z}) - 2\|\mathbf{X}\|_* \geq 0,$$

and the minimum value of $(1/2) \left[\mathrm{Tr}(\mathbf{Y}) + \mathrm{Tr}(\mathbf{Z}) \right]$ is $\|\mathbf{X}\|_*$. □

From simplex methods to interior point methods (IPM)

Observation

Simplex methods scans the extreme points of the feasible set, and this is why the number of iterations can be sub-exponential in the problem dimensions. (Although empirical performance is much better.)



Interior point method (IPM) [10, 12]

The key idea of the Interior Point Methods (IPM) is, as the name suggests, to keep the iterates in the interior of the feasible set, and progress toward the optimum.

A brief history of IPM

A brief history [10]

- N. Karmarkar proposed the first IPM for LP in 1984 [7].
  - This is the first algorithm for LP that has both theoretical polynomial time guarantee and good empirical performance.
- J. Renegar proposed the first path-following IPM for LP in 1986 [14].
  - This establishes the foundation of the current version of IPMs.
- Y. Nesterov extended the path-following idea to general constrained convex optimization problems in 1988 [12].
  - This is achieved by the notion of self-concordant barriers.

The path-following IPM

Let $\mathcal{G}$ be a closed bounded set in $\mathbb{R}^p$. Consider the convex program

$$\mathrm{opt} = \min_{\mathbf{x}} \left\{ \boldsymbol{c}^T \mathbf{x} : \mathbf{x} \in \mathcal{G} \right\}.$$

Definition (Barrier function)

A barrier function of $\mathcal{G}$ is a smooth convex function $F : \mathrm{int}(\mathcal{G}) \to \mathbb{R}$ such that $\lim_{t\to\infty} F(\mathbf{x}_t) = \infty$ for any sequence $\{\mathbf{x}_t : t \in \mathbb{N}\}$ converging to the boundary of $\mathcal{G}$, and $\nabla^2 F(\mathbf{x}) \succ \mathbf{0}$ for all $\mathbf{x} \in \mathrm{int}(\mathcal{G})$.

Idea of a path-following IPM

Consider a family of optimization problems:

$$x^*(t) = \arg\min_{\mathbf{x}} \left\{ t\boldsymbol{c}^T \mathbf{x} + F(\mathbf{x}) : \mathbf{x} \in \mathbb{R}^p \right\},$$

where F is a barrier function of $\mathcal{G}$. We call $x^*(t)$ the path.

- For every $t > 0$, $x^*(t)$ uniquely exists in $\mathcal{G}$, and hence is always feasible.
- For any sequence $t_i \to \infty$, we have $x^*(t_i) \to \mathrm{opt}$.
- For any sequence x_i such that $x_i - x^*(t_i) \to 0$, we have $x_i \to \mathrm{opt}$.

An example of the path-following IPM

Consider an SDP given by

$$\mathrm{opt} = \min_{\mathbf{x}} \left\{ \boldsymbol{c}^T \mathbf{x} : \mathbf{x} \in \mathbb{R}^p, \mathbf{F}_0 + \sum_{i=1}^{p} \mathbf{x}_i \mathbf{F}_i \succeq \mathbf{0} \right\},$$

for some $\boldsymbol{c} \in \mathbb{R}^p$ and symmetric matrices $\mathbf{F}_0, \ldots, \mathbf{F}_p \in \mathbb{R}^{m \times m}$.

Choice of the barrier function

The function $F(\mathbf{x}) := -\ln\det\left(\mathbf{F}_0 + \sum_{i=1}^{p} \mathbf{x}_i \mathbf{F}_i\right)$ is a (self-concordant) barrier function of the feasible set.

- Obviously, $F(\mathbf{x}) < +\infty$ if and only if $\mathbf{x}$ is in the feasible set of the SDP.

An example of the path-following IPM contd.

Define $F_t(\mathbf{x}) := t\boldsymbol{c}^T\mathbf{x} + F(\mathbf{x})$, and

$$\lambda(F_t, \mathbf{x}) := \sqrt{[\nabla F_t(\mathbf{x})]^T \left[\nabla^2 F(\mathbf{x})\right]^{-1} [\nabla F_t(\mathbf{x})]}.$$

It can be proved that $\mathbf{x}$ is close to $\mathbf{x}^*(t)$ if $\lambda(F_t, \mathbf{x})$ is small.

Basic path-following scheme [10]

Basic path-following scheme

Let $T \in \mathbb{N}$, and $\gamma, \kappa > 0$.

1. Set $t_0, \mathbf{x}_0$ such that $\lambda(t_0, \mathbf{x}_0) \leq \kappa$.

2. For $i = 1, \ldots, T$

$$t_i \leftarrow \left(1 + \frac{\gamma}{\sqrt{m}}\right) t_{i-1}$$

Find $\mathbf{x}_i$ such that $\lambda(F_{t_i}, \mathbf{x}_i) \leq \kappa$.

3. Output $\mathbf{x}_T$.

Theorem ([10])

The output of the basic path-following scheme satisfies $\mathbf{c}^T\mathbf{x}_T - \mathrm{opt} = \mathcal{O}(e^{-T})$.

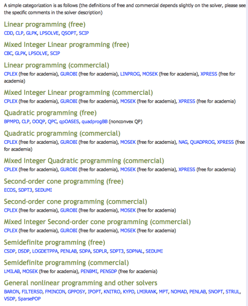
Available solvers

Warning

Kids, do not try this at home!
The following solvers have been designed by trained professionals.

Solvers

A list of solvers (commercial, academic free license and free/open source) categorized by application are available at



A simple categorization is as follows (the definitions of free and commercial depends slightly on the solver, please see the specific comments in the solver description)

Linear programming (free)
CDD, CLP, GLPK, LPSOLVE, QSOPT, SCIP

Mixed Integer Linear programming (free)
CBC, GLPK, LPSOLVE, SCIP

Linear programming (commercial)
CPLEX (free for academia), GUROBI (free for academia), LINPROG, MOSEK (free for academia), XPRESS (free for academia)

Mixed Integer Linear programming (commercial)
CPLEX (free for academia), GUROBI (free for academia), MOSEK (free for academia), XPRESS (free for academia)

Quadratic programming (free)
BPMPD, CLP, OOQP, QPC, qpOASES, quadprogBB (nonconvex QP)

Quadratic programming (commercial)
CPLEX (free for academia), GUROBI (free for academia), MOSEK (free for academia), NAG, QUADPROG, XPRESS (free for academia)

Mixed Integer Quadratic programming (commercial)
CPLEX (free for academia), GUROBI (free for academia), MOSEK (free for academia), XPRESS (free for academia)

Second-order cone programming (free)
ECOS, SDPT3, SEDUMI

Second-order cone programming (commercial)
CPLEX (free for academia), GUROBI (free for academia), MOSEK (free for academia)

Mixed Integer Second-order cone programming (commercial)
CPLEX (free for academia), GUROBI (free for academia), MOSEK (free for academia)

Semidefinite programming (free)
CSDP, DSDP, LOGDETPPA, PENLAB, SDPA, SDPLR, SDPT3, SDPNAL, SEDUMI

Semidefinite programming (commercial)
LMILAB, MOSEK (free for academia), PENBMI, PENSDP (free for academia)

General nonlinear programming and other solvers
BARON, FILTERSD, FMINCON, GPPOSY, IPOPT, KNITRO, KYPD, LMIRANK, MPT, NOMAD, PENLAB, SNOPT, STRUL, VSDP, SparsePOP

http://users.isy.liu.se/johanl/yalmip/pmwiki.php?n=Solvers.Solvers

References I

[1] Nina Amenta and Günter M. Ziegler.
Deformed products and maximal shadows of polytopes.
In *Advances in Discrete and Computational Geometry*, pages 57–90. AMS, 1999.

[2] Peter Bickel, Ya'acov Ritov, and Alexandre B. Tsybakov.
Simultaneous analysis of Lasso and Dantzig selector.
Ann. Stat., 37(4):1705–1732, 2009.

[3] Emmanuel Candes and Terence Tao.
The Dantzig selector: Statistical estimation when p is much larger than n.
Ann. Stat., 35(6):2313–2351, 2007.

[4] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders.
Atomic decomposition by basis pursuit.
SIAM J. Sci. Comput., 20(1):33–61, 1998.

[5] P. Elias, A. Feinstein, and C. E. Shannon.
A note on the maximum flow through a network.
IRE Trans. Inf. Theory, 2(4):117–119, 1956.

[6] Gil Kalai and Daniel J. Kleitman.
A quasi-polynomial bound for the diameter of graphs of polyhedra.
Bull. Amer. Math. Soc., 26(2):315–316, 1992.

References II

[7] N. Karmarkar.
A new polynomial-time algorithm for linear programming.
In *Proc. 16th Annu. ACM Symp. Theory of Computing*, pages 302–311, 1984.

[8] M. Krein and D. Milman.
On extreme points of regular convex sets.
Stud. Math., 9(1):133–138, 1940.

[9] Harry Markowtiz.
Portfolio selection.
J. Finance, 7(1):77–91, 1952.

[10] Arkadi Nemirovski.
Interior point polynomial time methods in convex programming.
2004.

[11] Arkadi Nemirovski.
Introduction to linear optimization.
2012.

[12] Yurii Nesterov and Arkadii Nemirovskii.
Interior-Point Polynomial Algorithms in Convex Programming.
SIAM, Philadelphia, PA, 1994.

References III

[13] Benjamin Recht, Maryam Fazel, and Pablo A. Parrilo.
Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization.
SIAM Rev., 52(3):471–501, 2010.

[14] James Renegar.
A polynomial-time algorithm, based on Newton's method, for linear programming.
Math. Program., 40:59–93, 1988.

[15] Steve Smale.
Mathematical problems for the next century.
Math. Intell., 20(2):7–15, 1998.

[16] Daniel A. Spielman and Shang-Hua Teng.
Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time.
J. ACM, 51(3):385–463, 2004.

[17] Lieven Vandenberghe and Stephen Boyd.
Semidefinite programming.
SIAM Rev., 38(1):49–95, 1996.

References IV

[18] Vijay V. Vazirani.
Approximation Algorithms, chapter 12, pages 93–107.
Springer, Berlin, 2003.