LMI Methods in Optimal and Robust Control

Matthew M. Peet
Arizona State University

Lecture 02: Optimization (Convex and Otherwise)

Mathematical Optimization and Curly's Law

Curly: Do you know what the secret of life is?
Curly: One thing (metric). Just one thing. You stick to that (metric) and the rest don't mean ****.

$$\min_{x\in\mathbb{F}} \quad f(x): \qquad \text{subject to}$$
$$g_i(x) \le 0 \qquad i = 1, \cdots K_1$$
$$h_i(x) = 0 \qquad i = 1, \cdots K_2$$

Variables: $x \in \mathbb{F}$

- The things you must choose.
- $\mathbb{F}$ represents the set of possible choices for the variables.
- Can be vectors, matrices, functions, systems, locations, colors...
  - However, computers prefer vectors or matrices.

Objective: $f(x)$

- A function which assigns a *scalar* value to any choice of variables.
  - e.g. $[x_1, x_2] \mapsto x_1 - x_2$; red $\mapsto 4$; et c.

Constraints: $g(x) \le 0$; $h(x) = 0$

- Defines what is a minimally acceptable choice of variables (Feasible).

Mathematical Optimization and Curly's Law

Curly: Do you know what the secret of life is?
Curly: One thing (metric). Just one thing. You stick to that (metric) and the rest don't mean ****.

$$\min_{x\in\mathbb{F}} \quad f(x): \qquad \text{subject to}$$
$$g_i(x) \le 0 \qquad i = 1, \cdots K_1$$
$$h_i(x) = 0 \qquad i = 1, \cdots K_2$$

Variables: $x \in \mathbb{F}$
- The things you must choose.
- $\mathbb{F}$ represents the set of possible choices for the variables.
- Can be vectors, matrices, functions, systems, locations, colors...
  - However, computers prefer vectors or matrices.

Objective: $f(x)$
- A function which assigns a *scalar* value to any choice of variables.
  - e.g. $[x_1, x_2] \mapsto x_1 - x_2$; red $\mapsto 4$; et c.

Constraints: $g(x) \le 0$; $h(x) = 0$
- Defines what is a minimally acceptable choice of variables (Feasible).

"All [mankind] . . . are endowed by their Creator with certain unalienable Rights, that among these are Life, Liberty and the pursuit of Happiness." - Thomas Jefferson, Declaration of Independence (July 4, 1776)

"Happiness is the meaning and the purpose of life, the whole aim and end of human existence" – Aristotle (384-322 BC)

"Happiness, private happiness, is the proper or ultimate end of all our action" – John Gay, Concerning the Fundamental Principle of Virtue or Morality (1731)

"this circumstance of public utility is ever principally in view; and wherever disputes arise, either in philosophy or common life, concerning the bounds of duty, the question cannot, by any means, be decided with greater certainty, than by ascertaining, on any side, the true interests of mankind." – David Hume, An Enquiry Concerning the Principles of Morals (1751)

"You wake up at 3:00 in the morning and say oh my god, everything is an optimization problem . . . Get over it quickly, please. Of course, everything is an optimization problem." – Steven Boyd

Mathematical Optimization

Why Optimization:

- Because its **Optimal** Control!
- Optimization algorithms are the most advanced and universal computational tool we have.
  - But they can't solve every problem

Topics to Cover:

Mathematical Optimization

- Objective, variables, constraints
- Different formulations of the same problem

Convex Optimization

- What is convex optimization?
- Why is it important?

Computational Complexity

- Computational Complexity
- How hard are optimization problems to solve?

Goals: Given an optimization problem

- Be able to identify variables, constraints, and objective
- Know if an optimization problem is convex

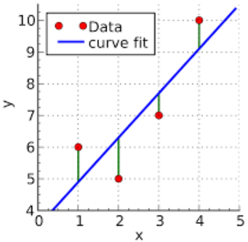
Linear Least Squares (Early Machine Learning)

The First Mathematical Form of Optimization

Problem: Given a bunch of data in the form

- Inputs: a_i
- Outputs: b_i

Find the function $f(a) = b$ which best fits the data.



For **Least Squares**: Assume $f(a) = z^T a + z_0$ where $z \in \mathbb{R}^n, z_0 \in \mathbb{R}$ are the variables with objective

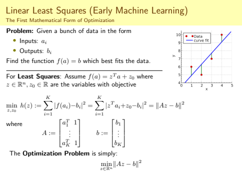
$$\min_{z,z_0} \; h(z) := \sum_{i=1}^{K} |f(a_i) - b_i|^2 = \sum_{i=1}^{K} |z^T a_i + z_0 - b_i|^2 = \|Az - b\|^2$$

where

$$A := \begin{bmatrix} a_1^T & 1 \\ & \vdots \\ a_K^T & 1 \end{bmatrix} \qquad b := \begin{bmatrix} b_1 \\ \vdots \\ b_K \end{bmatrix}$$

The **Optimization Problem** is simply:

$$\min_{z \in \mathbb{R}^n} \|Az - b\|^2$$



Linear Least Squares (Early Machine Learning)
The First Mathematical Form of Optimization

Problem: Given a bunch of data in the form
- Inputs: a_i
- Outputs: b_i

Find the function $f(a) = b$ which best fits the data.

For **Least Squares**: Assume $f(a) = z^T a + z_0$ where $z \in \mathbb{R}^n, z_0 \in \mathbb{R}$ are the variables with objective

$$\min_{z,z_0} h(z) := \sum_{i=1}^{K} |f(a_i) - b_i|^2 = \sum_{i=1}^{K} |z^T a_i + z_0 - b_i|^2 = \|Az - b\|^2$$

where
$$A := \begin{bmatrix} a_1^T & 1 \\ & \vdots \\ a_K^T & 1 \end{bmatrix} \qquad b := \begin{bmatrix} b_1 \\ \vdots \\ b_K \end{bmatrix}$$

The **Optimization Problem** is simply:
$$\min_{z \in \mathbb{R}^n} \|Az - b\|^2$$

Boring/Conservative/Grumpy (Monarchist).

One of the greatest mathematicians
- Professor of Astronomy in Göttingen
- Motto: "pauca sed matura" (few but ripe)

Discovered
- Gaussian Distributions
- Gauss' Law (collaboration with Weber)
- Non-Euclidean Geometry (maybe)
- Least Squares (maybe)

Legendre published the first solution to the Least Squares problem in 1805
- In typical fashion, Carl Friedrich Gauss claimed to have solved the problem in 1795 and published a more rigorous solution in 1809.
- This more rigorous solution first introduced the normal probability distribution (or Gaussian distribution)

Solution to the Least Squares Problem

An **Unconstrained** Optimization Problem

The **Least Squares Problem** is:

$$\min_{z\in\mathbb{R}^n} h(z) := \|Az-b\|^2$$

where

$$A := \begin{bmatrix} a_1^T & 1 \\ & \vdots \\ a_K^T & 1 \end{bmatrix} \qquad b := \begin{bmatrix} b_1 \\ \vdots \\ b_K \end{bmatrix}$$

Least squares problems are easy-ish to solve.

- If z^* minimizes $h(z)$, then $\nabla_z h(z^*) = 0$

$$\nabla_z h(z^*) = 0 \qquad \leftrightarrow \qquad z^* = (A^TA)^{-1}A^Tb$$

Note that A is assumed to be skinny.

- More rows than columns (i.e. More data points than inputs).
- Don't need YALMIP to solve this one...

In *Classification* we have inputs (data) (x_i), each of which has a binary label $(y_i \in \{-1, +1\})$

- $y_i = +1$ means the output of x_i belongs to group 1
- $y_i = -1$ means the output of x_i belongs to group 2

We want to find a rule (a classifier) which takes the data x and predicts which group it is in.

- Our rule has the form of a function $f(x) = w^T x - b$. Then
  - x is in group 1 if $f(x) = w^T x - b > 0$.
  - x is in group 2 if $f(x) = w^T x - b < 0$.

Question: How to find the best w and b??

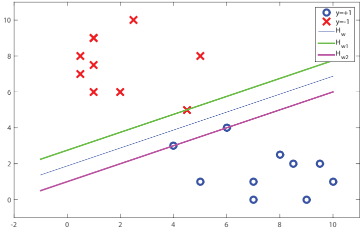
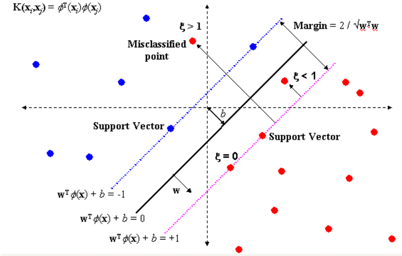


Figure: We want to find a rule which separates two sets of data.



Definition 1.

- A **Hyperplane** is the generalization of the concept of line/plane to multiple dimensions.
$$\{x \in \mathbb{R}^n \;:\; w^T x - b = 0\}$$
- **Half-Spaces** are the parts above and below a Hyperplane.
$$\{x \in \mathbb{R}^n \;:\; w^T x - b \geq 0\} \qquad \text{OR} \qquad \{x \in \mathbb{R}^n \;:\; w^T x - b \leq 0\}$$

We want to separate the data into disjoint half-spaces and maximize the distance between these half-spaces

Variables: $w \in \mathbb{R}^n$ and b define the hyperplane
Constraint: Each datum has the right label.

- $w^T x - b > 1$ when $y_i = +1$ and
  $w^T x - b < -1$ when $y_i = -1$
- Alternatively: $y_i(w^T x_i - b) \geq 1$.

These two constraints are **Equivalent**.

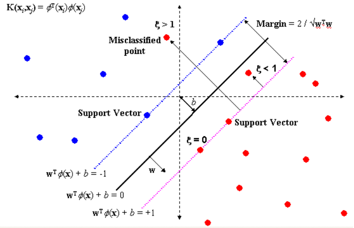


Figure: Maximizing the distance between two sets of Data

Objective: The distance between Hyperplanes $\{x \,:\, w^T x - b = 1\}$ and $\{x \,:\, w^T x - b = -1\}$ is

$$f(w, b) = 2\frac{1}{\sqrt{w^T w}}$$

Machine Learning

Hard and Soft Margin SVM

Hard Margin SVM solves

$$\min_{w\in\mathbb{R}^p, b\in\mathbb{R}} \frac{1}{2} w^T w, \qquad \text{subject to}$$

$$y_i(w^T x_i - b) \geq 1, \quad \forall i = 1, ..., K.$$

Soft Margin SVM

The hard margin problem can be relaxed to maximize the distance between hyperplanes PLUS the magnitude of classification errors

$$\min_{w\in\mathbb{R}^p, b\in\mathbb{R}} \frac{1}{2}\|w\|^2 + c\sum_{i=1}^{n} \max(0, 1-(w^T x_i - b)y_i).$$

This is unconstrained optimization

- Constraints become penalties!

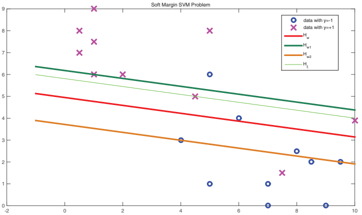


Figure: Data separation using soft-margin metric and distances to associated hyperplanes

Link: Repository of Interesting Machine Learning Data Sets

MAX-CUT: A *Classic* Integer Programming Example

binary variables and 'or' constraints

Optimization of *Nodes* and *Edges*.

- We want to assign each node to group 1 or Group 2.
- We get paid w_{ij} dollars for putting node i and node j in different groups.

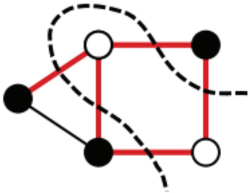


Figure: Division of a set of nodes to maximize the weighted cost of separation

Goal: Assign each node i an *index* $x_i = -1$ or $x_i = 1$ to maximize profit.

- The profit if x_i and x_j do not share the same index is w_{ij}.
- No profit if they share an index is 0
- The weights w_{ij} are all known.

MAX-CUT

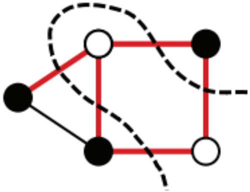
What are the decision variables?

Goal: Assign each node i an index $x_i = -1$ or $x_j = 1$ to maximize overall cost.

Variables: $x \in \{-1, 1\}^n$

- Referred to as **Integer Variables** or **Binary Variables**.
- Binary constraints can be incorporated explicitly:

$$x_i^2 = 1$$

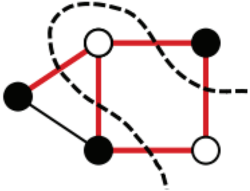


Integer/Binary variables may be declared directly in YALMIP:

```
> x = intvar(n);
> y = binvar(n);
```

Question: How do we use integer variables to define the objective?

Answer: We use the trick:



- $(1 - x_i x_j) = 0$ if x_i and x_j have the same sign (Together).
- $(1 - x_i x_j) = 2$ if x_i and x_j have the opposite sign (Apart).

Then the objective function is

$$\min \frac{1}{2} \sum_{i,j} w_{ij}(1 - x_i x_j)$$

The optimization problem is the integer program:

$$\max_{x_i^2 = 1} \frac{1}{2} \sum_{i,j} w_{ij}(1 - x_i x_j)$$

MAX-CUT

The optimization problem is the integer program:

$$\max_{x_i^2=1} \frac{1}{2} \sum_{i,j} w_{ij}(1 - x_i x_j)$$

Consider the MAX-CUT problem with 5 nodes

$$w_{12} = w_{23} = w_{45} = w_{15} = w_{34} = .5 \quad \text{and} \quad w_{14} = w_{24} = w_{25} = 0$$

where $w_{ij} = w_{ji}$.

An Optimal Cut IS:

- $x_1 = x_3 = x_4 = 1$
- $x_2 = x_5 = -1$

This cut has objective value

$$f(x) = 2.5 - .5x_1x_2 - .5x_2x_3 - .5x_3x_4 - .5x_4x_5 - .5x_1x_5 = 4$$

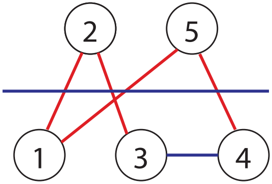


Figure: An Optimal Cut

The Knapsack Problem

Formulate the Optimization Problem

Suppose you have a knapsack (backpack), into which you can fit 12kg of stuff.

- You are given a list of possible items to place in the knapsack, along with the value (in \$) of each item.
- For Example:

Item	Weight (kg)	Value (\$)
gloves	1	4
coat	13	200
pants	6	4
underthings	1	2
socks	1	1
shirt	5	2

- The **objective** is to stuff as much value into the knapsack as possible.

Question: How to formulate the mathematical optimization problem?

Optimization with Dynamics

Open-Loop Case (aka Dynamic Programming or Model Predictive Control)

Objective Function: Lets minimize a quadratic cost

$$x(N)^T S x(N) + \sum_{k=1}^{N-1} x(k)^T Q x(k) + u(k)^T R u(k)$$

Variables: The sequence of states $x(k)$, and inputs, $u(k)$.
Constraint: The dynamics define how $u \mapsto x$.

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \qquad k = 0, \cdots, N \\ x(0) &= 1 \end{aligned}$$

Optimization Formulation of DP:

$$\min_{x,u} \quad x(N)^T S x(N) + \sum_{k=1}^{N-1} \left(x(k)^T Q x(k) + u(k)^T R u(k) \right)$$

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \qquad k = 0, \cdots, N \\ x(0) &= 1 \end{aligned}$$

Optimization with Dynamics

Open-Loop Case (aka Dynamic Programming or Model Predictive Control)

Objective Function: Lets minimize a quadratic cost

$$x(N)^T S x(N) + \sum_{k=1}^{N-1} x(k)^T Q x(k) + u(k)^T R u(k)$$

Variables: The sequence of states $x(k)$, and inputs, $u(k)$.
Constraint: The dynamics define how $u \mapsto x$.

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \qquad k = 0, \cdots, N \\ x(0) &= 1 \end{aligned}$$

Optimization Formulation of DP:

$$\begin{aligned} \min_{x,u} \quad & x(N)^T S x(N) + \sum_{k=1}^{N-1} \left(x(k)^T Q x(k) + u(k)^T R u(k) \right) \\ x(k+1) &= Ax(k) + Bu(k), \qquad k = 0, \cdots, N \\ x(0) &= 1 \end{aligned}$$

Dynamic Programming has been around since the 1950's and can be solved recursively using Bellman's equation

- Actually, a nested sequence of optimization problems.
- Solution relies on the "Principle of Optimality"
- The principle of Optimality says that if we start anywhere along the optimal trajectory, that solution will still be optimal if we re-started the optimization problem from that point.
- Implies that the optimal input (for separable objectives) is always a function of the current state.
- The principle of optimality is also what underlies Djikstra's algorithm
- Djikstra's algorithm is what enables internet packet routing and the route-finding in Google (Apple) maps.

Objective Function: Lets minimize a quadratic Cost

$$x(N)^T S x(N) + \sum_{k=1}^{N-1} x(k)^T Q x(k) + u(k)^T R u(k)$$

Variables: We want to find the gain matrix, K, so that $u(k) = Kx(k)$.
Constraint: The dynamics define how $u \mapsto x$.

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \qquad k = 0, \cdots, N \\ u(k) &= Kx(k), \qquad x(0) = 1 \end{aligned}$$

Optimization Formulation of LQR:

$$\min_K \quad x(N)^T S x(N) + \sum_{k=1}^{N-1} \left(x(k)^T Q x(k) + u(k)^T R u(k) \right)$$

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \qquad k = 0, \cdots, N \\ u(k) &= Kx(k), \qquad x(0) = 1 \end{aligned}$$

Question: Are the Closed-Loop and Open-Loop Problems Equivalent?

Optimization with Dynamics

Closed-Loop Case (LQR)

Objective Function: Lets minimize a quadratic Cost

$$x(N)^T S x(N) + \sum_{k=1}^{N-1} x(k)^T Q x(k) + u(k)^T R u(k)$$

Variables: We want to find the gain matrix, K, so that $u(k) = Kx(k)$.
Constraint: The dynamics define how $u \mapsto x$.

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \qquad k = 0, \cdots, N \\ u(k) &= Kx(k), \qquad x(0) = 1 \end{aligned}$$

Optimization Formulation of LQR:

$$\min_K \quad x(N)^T S x(N) + \sum_{k=1}^{N-1} \left(x(k)^T Q x(k) + u(k)^T R u(k) \right)$$

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \qquad k = 0, \cdots, N \\ u(k) &= Kx(k), \qquad x(0) = 1 \end{aligned}$$

Question: Are the Closed-Loop and Open-Loop Problems Equivalent?

LQR stands for Least Quadratic Regulator

- Least Quadratic refers to the quadratic cost function
- Regulator refers to feedback

By Equivalent, can we assume the optimal input is a static function of the current state?

- Bellman's equation says the optimal input for separable objectives is always a function of the current state.
- In the quadratic case, the resulting function is static

Definition 2.

Two optimization problems are **Equivalent** if a solution (algorithm/black box) to one can be used to construct a solution to the other.

Trick 1: Equivalent Objective Functions

Problem 1: $\min_x f(x)$ subject to $A^T x \geq b$

Problem 2: $\min_x 10f(x) - 12$ subject to $A^T x \geq b$

Problem 3: $\max_x \frac{1}{f(x)}$ subject to $A^T x \geq b$

In this case $x_1^* = x_2^* = x_3^*$. Proof:

- For any $x \neq x_1^*$ (both feasible), since x_1^* is optimal, we have $f(x) > f(x_1^*)$. Thus $10f(x) - 12 > 10f(x_1^*) - 12$ and $\frac{1}{f(x)} < \frac{1}{f(x_1^*)}$. i.e x is suboptimal for all.

Formulating Optimization Problems

Equivalence in Variables

Trick 2: Invertible Change of Variables

Problem 1:	$\min_y f(y)$	subject to	$A^T y \ge b$
Problem 2:	$\min_x f(Tx+c)$	subject to	$(T^T A)^T x \ge b - A^T c$

Here $y^* = Tx^* + c$ and $x^* = T^{-1}(y^* - c)$.

- Proof hint: Given $y \neq Tx^* + c$, show $f(y) > f(Tx^* + c)$.

Trick 3: Variable Separability (e.g. Dynamic Programming)

Problem 1:	$\min_{x,y} f(x) + g(y)$	subject to	$A_1^T x \ge b_1, A_2^T y \ge b_2$
Problem 2:	$\min_w f(w)$	subject to	$A_1^T w \ge b_1$
Problem 3:	$\min_z g(z)$	subject to	$A_2^T z \ge b_2$

Then $x^* = w^*$ and $y^* = z^*$.

- Neither feasibility nor minimality are coupled (Objective fn. is *Separable*).

Trick 4: Constraint/Objective Equivalence

Problem 1: $\min_{x} f(x)$ subject to $g(x) \leq 0$

Problem 2: $\min_{y,t} t$ subject to $g(y) \leq 0, \; t \geq f(y)$

Here $y^* = x^*$ and $t^* = f(x^*)$.

Some other Equivalences:

- Redundant Constraints
  - $\{x \in \mathbb{R} : \; x > 1\}$ vs. $\{x \in \mathbb{R} : \; x > 1, x > 0\}$
- Polytopes (Vertices vs. Hyperplanes)
  - $\{x \in \mathbb{R}^n : \; x = \sum_i A_i \alpha_i, \sum_i \alpha_i = 1\}$ vs. $\{x \in \mathbb{R}^n : \; Cx > b\}$

Conclusion: Many different ways to represent the same optimization problem.

These Problems are all equivalent:

The **Classical Representation**:

$$\min_{x\in\mathbb{R}^n} f(x) : \qquad \text{subject to}$$

$$g_i(x) \le 0 \qquad i = 1, \cdots k$$

The **Geometric Representation** is:

$$\min_{x\in\mathbb{R}^n} \; f(x) : \qquad \text{subject to} \qquad x \in S$$

where $S := \{x \in \mathbb{R}^n \,:\, g_i(x) \le 0,\; i = 1, \cdots, k\}$.

The **Pure Geometric Representation** (x is eliminated!):

$$\min_{\gamma} \; \gamma : \qquad \text{subject to}$$

$$S_\gamma \neq \emptyset \quad (S_\gamma \text{ has at least one element})$$

where $S_\gamma := \{x \in \mathbb{R}^n \,:\, \gamma - f(x) \ge 0,\, g_i(x) \le 0,\; i = 1, \cdots, k\}$.

Proposition: Optimization is only as hard as determining feasibility!

Geometric vs. Functional Constraints

General Theory of Equivalent Optimization Problems

These Problems are all equivalent:

The **Classical Representation**:

$$\min_{x\in\mathbb{R}^n} f(x): \quad \text{subject to}$$
$$g_i(x) \le 0 \qquad i = 1, \cdots k$$

The **Geometric Representation** is:

$$\min_{x\in\mathbb{R}^n} f(x): \quad \text{subject to} \quad x \in S$$

where $S := \{x \in \mathbb{R}^n : g_i(x) \le 0,\ i = 1, \cdots, k\}$.

The **Pure Geometric Representation** (x is eliminated!):

$$\min_{\gamma} \gamma: \quad \text{subject to}$$
$$S_\gamma \neq \emptyset \quad (S_\gamma \text{ has at least one element})$$

where $S_\gamma := \{x \in \mathbb{R}^n : \gamma - f(x) \ge 0,\ g_i(x) \le 0,\ i = 1, \cdots, k\}$.

Proposition: Optimization is only as hard as determining feasibility!

In the pure geometric interpretation, we are finding the smallest γ such that there exists a feasible point, x with $f(x) \le \gamma$

Bisection: Optimization by testing feasibility

Power of the magic 8-ball

Optimization Problem:

$$\gamma^* = \max_{\gamma} \; \gamma :$$

$$\text{subject to } S_\gamma \neq \emptyset$$

Bisection Algorithm (Convexity???):

1 Initialize infeasible $\gamma_u = b$

2 Initialize feasible $\gamma_l = a$

3 Set $\gamma = \frac{\gamma_u + \gamma_l}{2}$

5 If S_γ feasible, set $\gamma_l = \frac{\gamma_u + \gamma_l}{2}$

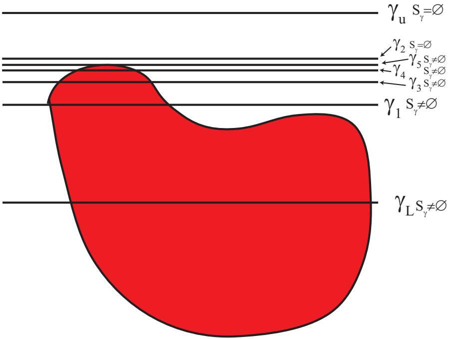
4 If S_γ infeasible, set $\gamma_u = \frac{\gamma_u + \gamma_l}{2}$

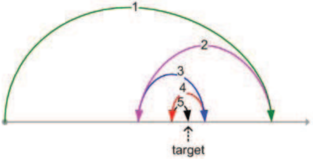
6 $k = k + 1$

7 Goto 3

Then $\gamma^* \in [\gamma_l, \gamma_u]$ and $|\gamma_u - \gamma_l| \leq \frac{b-a}{2^k}$.

Bisection with oracle also solves the Primary Problem. ($\min \gamma : \; S_\gamma = \emptyset$)





How Hard is an Optimization Problem?

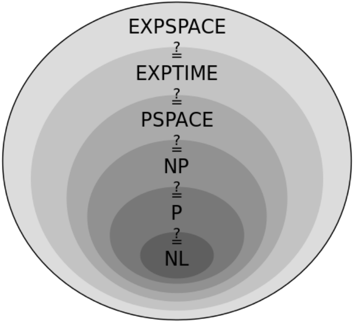
We define "Hard" using Computational Complexity

In Computer Science, we focus on **Complexity of the PROBLEM**

- NOT complexity of the algorithm.

On a Turing machine, the # of steps is a fn of problem size (number of variables)

- NL: A logarithmic # (SORT)
- P: A polynomial # (LP)
- NP: A polynomial # for verification (TSP)
- NP HARD: at least as hard as NP (TSP)
- NP COMPLETE: A set of Equivalent* NP problems (MAX-CUT, TSP)
- EXPTIME: Solvable in $2^{p(n)}$ steps. p polynomial. (Chess)
- EXPSPACE: Solvable with $2^{p(n)}$ memory.



*Equivalent means there is a polynomial-time reduction from one to the other.

How Hard is Optimization?

The **Classical Representation**:

$$\min_{x\in\mathbb{R}^n} f(x): \qquad \text{subject to}$$
$$g_i(x) \le 0 \qquad i = 1, \cdots k$$
$$h_i(x) = 0 \qquad i = 1, \cdots k$$

Answer: Easy (P) if f, g_i are all Convex and h_i are affine.

The **Geometric Representation**:

$$\min_{x\in\mathbb{R}^n} \; f(x): \qquad \text{subject to} \qquad x \in S$$

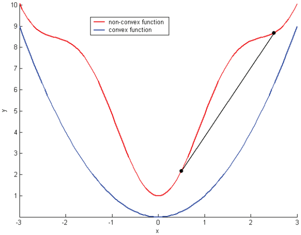
Answer: Easy (P) if f is Convex and S is a Convex Set.

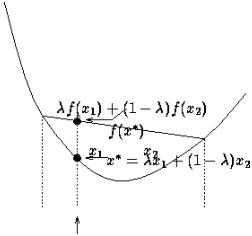
The **Pure Geometric Representation**:

$$\max_{\gamma, x\in\mathbb{R}^n} \; \gamma: \qquad \text{subject to}$$
$$(\gamma, x) \in S'$$

Answer: Easy (P) if S' is a Convex Set.

Convex Functions





Definition 3 (Convexity of a Function).

A **Function** (objective or constraint) is **Convex** if
$f(\lambda x_1 + (1-\lambda)x_2) \le \lambda f(x_1) + (1-\lambda) f(x_2)$ for all $\lambda \in [0,1]$.

Useful Facts:

- e^{ax}, $\|x\|$ are convex. x^n ($n \ge 1$ or $n \le 0$), $-\log x$ are convex on $x \ge 0$
- If f_1 is convex and f_2 is convex, then $f_3(x) := f_1(x) + f_2(x)$ is convex.
- A f is convex iff the Hessian $\nabla^2 f(x)$ is positive semidefinite for all x.
- If f_1, f_2 are convex, then $f_3(x) := \max(f_1(x), f_2(x))$ is convex.
- If f_1, f_2 are convex, and f_1 is increasing, then $f_3(x) := f_1(f_2(x))$ is convex.

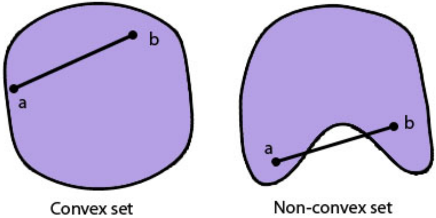
Convex Sets

Definition 4 (Convexity of a Set).

A **Set** (of feasible points) is **Convex** if for any $x, y \in Q$,

$$\{\mu x + (1-\mu)y \,:\, \mu \in [0,1]\} \subset Q.$$

The line connecting any two points lies in the set.



Facts:

- If function g is convex, the feasible set, $S = \{x : g(x) \le 0\}$, is also convex.
- The intersection of convex sets is convex.
  - If S_1 and S_2 are convex, then $S_2 := \{x \,:\, x \in S_1,\ x \in S_2\}$ is convex.

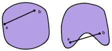
Convex Sets

Definition 4 (Convexity of a Set).

A **Set** (of feasible points) is **Convex** if for any $x, y \in Q$,

$$\{\mu x + (1-\mu)y \,:\, \mu \in [0,1]\} \subset Q.$$

The line connecting any two points lies in the set.



Convex set Non-convex set

Facts:

- If function g is convex, the feasible set, $S = \{x : g(x) \leq 0\}$, is also convex.
- The intersection of convex sets is convex.
  - If S_1 and S_2 are convex, then $S_2 := \{x \,:\, x \in S_1,\ x \in S_2\}$ is convex.

An optimization problem is convex if the objective and set of feasible points are both convex.

Pop Quiz:

$$\min_{x,y \in \mathbb{R}^2} \quad f(x,y) = x^2 - 3xy + y^2 :$$
$$g(x,y) = 1 - x^2 - y^2 \geq 0$$
$$h(x,y) = x - y = 0$$

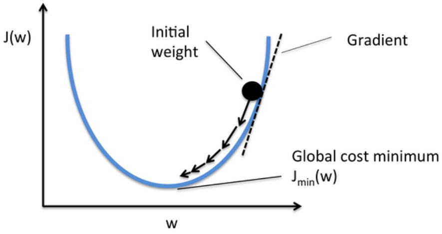
Question: Is this convex optimization?

All descent algorithms are iterative, with a search direction ($\Delta x \in \mathbb{R}^n$) and step size ($t \geq 0$).

$$x_{k+1} = x_k + t\Delta x$$

Gradient Descent

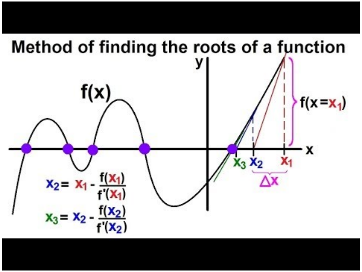
$$\Delta x = -\nabla f(x)$$



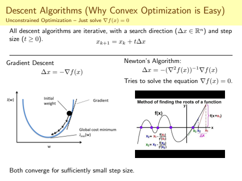
Newton's Algorithm:

$$\Delta x = -(\nabla^2 f(x))^{-1} \nabla f(x)$$

Tries to solve the equation $\nabla f(x) = 0$.



Both converge for sufficiently small step size.



- If $\nabla f(x) = 0$, then f has a minimum or maximum at x.
- In unconstrained optimization, the solution will occur at this inflection point.
- For a convex function, there is only one point where $\nabla f(x) = 0$, which is the global minimum.
- For constrained problems, we have KKT conditions – see the lecture notes...

Method 1: Gradient Projection

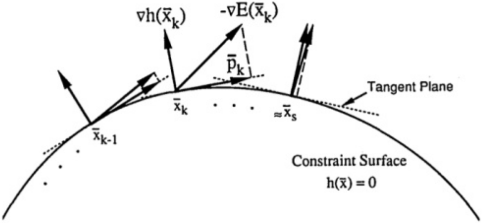


Figure: Must project step ($t\Delta x$) onto feasible Set

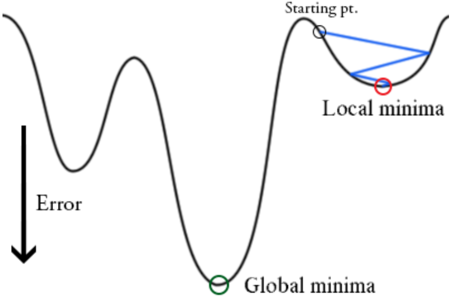
Method 2: Barrier Functions

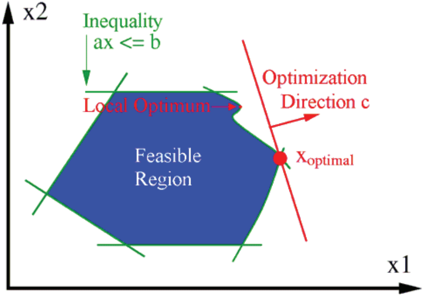
$$\min_x f(x) + \log(g(x))$$

Converts a Constrained problem to an unconstrained problem.
(Interior-Point Methods)

Non-Convexity and Local Optima

1. For convex optimization problems, Descent Methods always find the global optimal point.
2. For non-convex optimization, Descent Algorithms may get stuck at local optima.





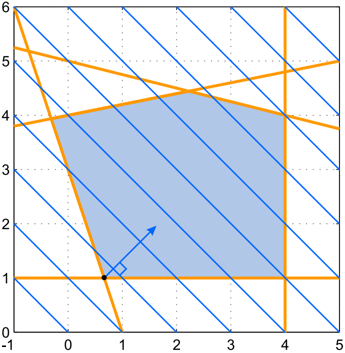
Linear Programming (LP)

$$\min_{x\in\mathbb{R}^n} c^T x : \qquad \text{subject to}$$

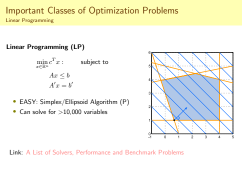
$$Ax \le b$$

$$A'x = b'$$

- EASY: Simplex/Ellipsoid Algorithm (P)
- Can solve for >10,000 variables



Link: A List of Solvers, Performance and Benchmark Problems



- The Ellipsoidal algorithm solves LP in polynomial time.
- The Simplex algorithm is not actually worst-case polynomial time.
- However, the simplex algorithm outperforms the ellipsoidal algorithm in almost all cases.

Quadratic Programming (QP)

$$\min_{x\in\mathbb{R}^n} x^TQx + c^Tx : \qquad \text{subject to}$$
$$Ax \le b$$

Quadratically Constrained Quadratic Programming (QCQP)

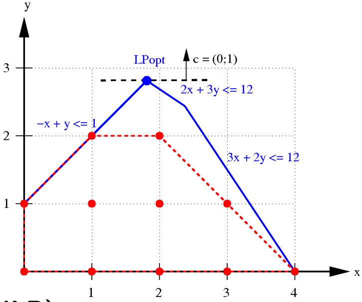
$$\min_{x\in\mathbb{R}^n} x^TQx + c^Tx : \qquad \text{subject to}$$
$$x^TPx + d^Tx \le f$$

- EASY (P): If $Q, P \ge 0$.
- HARD (NP-Hard): Otherwise

Mixed-Integer Linear Programming (MILP)

$$\min_{x \in \mathbb{R}^n} c^T x : \qquad \text{subject to}$$
$$Ax \le b$$
$$x_i \in \mathbb{Z} \qquad i = 1, \cdots K$$

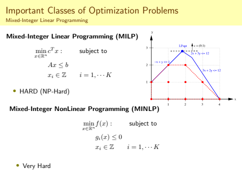
- HARD (NP-Hard)



Mixed-Integer NonLinear Programming (MINLP)

$$\min_{x \in \mathbb{R}^n} f(x) : \qquad \text{subject to}$$
$$g_i(x) \le 0$$
$$x_i \in \mathbb{Z} \qquad i = 1, \cdots K$$

- Very Hard



- Gurobi will allow you to solve very large MILPs.
- However, the result may not be truly optimal.

Pop Quiz: Classify the following optimization problems

$$\min_{x,y \in \mathbb{R}^2} \quad x^2 - 3xy + y^2 : \qquad 1 - x^2 - y^2 \geq 0, \qquad h(x,y) = x - y = 0$$

$$\min_{x,y \in \mathbb{R}^2} \quad x + y : \qquad x + 3y \leq 4, \qquad 4 - 3x \geq -1$$

$$\min_{x,y \in \{-1,0,1\}} \quad x + y : \qquad x + 3y \leq 4, \qquad 4 - 3x \geq -1$$

$$\min_{x,y \in \{-1,0,1\}} \quad x^2 - 3xy + y^2 : \qquad 1 - x^2 - y^2 \geq 0, \qquad h(x,y) = x - y = 0$$

Question: which are convex?

Original Problem:

$$\gamma^* := \min_{x \in \mathbb{R}} \quad f(x) : \qquad g(x) \geq 0 \;\; (FS)$$

Definition 5.

In a **Relaxation**, we remove or loosen one of the constraints.

$$\gamma_R^* := \min_{x \in \mathbb{R}} \quad f(x) : \qquad g(x) \geq -1$$

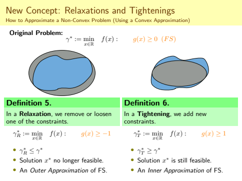
- $\gamma_R^* \leq \gamma^*$
- Solution x^* no longer feasible.
- An *Outer Approximation* of FS.

Definition 6.

In a **Tightening**, we add new constraints.

$$\gamma_T^* := \min_{x \in \mathbb{R}} \quad f(x) : \qquad g(x) \geq 1$$

- $\gamma_T^* \geq \gamma^*$
- Solution x^* is still feasible.
- An *Inner Approximation* of FS.



- FS stands for Feasible Set
  - The set of values of x which satisfy the constraints
- Relaxations *Increase* the size of the feasible set
  - The solution may not be feasible for the original problem
- Tightenings *Decrease* the size of the feasible set
  - The solution may not be optimal for the original problem

MAX-CUT: Original Problem

$$\max_{x_i^2=1} \frac{1}{2} \sum_{i,j} w_{i,j}(1 - x_i x_j)$$

Solution: $\gamma^* = 4$

- $x_1 = x_3 = x_4 = 1$
- $x_2 = x_5 = -1$

MAX-CUT: Relaxed Problem

$$\max_{x_i^2\le 1} \frac{1}{2} \sum_{i,j} w_{i,j}(1 - x_i x_j)$$

Solution: $\gamma_R^* = 4$

- $x_2 = x_5 = 1$
- $x_1 = x_4 = -1$
- $x_3 = 0$

YALMIP Code:

```
> x = sdpvar(5,1);
> F=[-1 <= x <= 1];
> obj=2.5-.5*x(1)*x(2)-.5*x(2)*x(3)
>         -.5*x(3)*x(4)-.5*x(4)*x(5)-.5*x(1)*x(5);
> optimize(F,-obj);
> value(x);
```

Link: Download CPLEX (Need version 12.10 or earlier)

Link: Download GUROBI

Relaxations and Tightenings

Examples

MAX-CUT: Original Problem

$$\max_{x_i^2=1} \frac{1}{2} \sum_{i,j} w_{i,j}(1 - x_i x_j)$$

Solution: $\gamma^* = 4$

- $x_1 = x_3 = x_4 = 1$
- $x_2 = x_5 = -1$

MAX-CUT: Relaxed Problem

$$\max_{x_i^2\le 1} \frac{1}{2} \sum_{i,j} w_{i,j}(1 - x_i x_j)$$

Solution: $\gamma_R^* = 4$

- $x_2 = x_5 = 1$
- $x_1 = x_4 = -1$
- $x_3 = 0$

YALMIP Code:

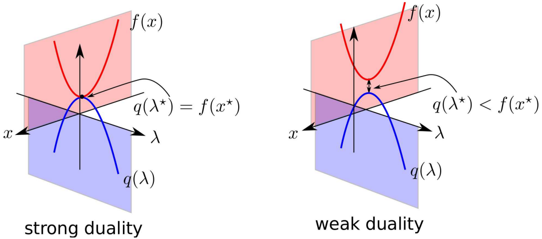
```
> x = sdpvar(5,1);
> F=[-1 <= x <= 1];
> obj=2.5-.5*x(1)*x(2)-.5*x(2)*x(3)
>       -.5*x(3)*x(4)-.5*x(4)*x(5)-.5*x(1)*x(5);
> optimize(F,-obj);
> value(x);
```

Link: Download CPLEX (Need version 12.10 or earlier)
Link: Download GUROBI

Note the solution to the relaxed Max Cut problem is not feasible for the original problem.

A Third Option: Duality

A Cool Word, but Meaning is Vague



Definition 7.

Two **Optimization Problems are Dual** if any feasible solution to one has objective value which bounds the solution to the other problem.

Primal Problem:

$$\min_{x \in \mathbb{R}} \quad f(x) : \qquad x \in S$$

Dual Problem:

$$\max_{y \in \mathbb{R}} \quad f_D(y) : \qquad y \in S_D$$

Relationship:

- if $y \in S_D$, then $f_D(y) \le f(x)$ for any $x \in S$.
- if $x \in S$, then $f(x) \ge f_D(y)$ for any $y \in S_D$.

Lagrangian Duality

$$\min_{x\in\mathbb{F}} f_0(x) : \qquad \text{subject to } f_i(x) \geq 0 \qquad i = 1, \cdots k$$

Note that

$$\max_{\alpha>0} -\alpha f_i(x) = \begin{cases} \infty & f_i(x) < 0 \\ 0 & \text{otherwise} \end{cases}$$

Equivalent Form:

$$\gamma^* = \min_{x\in\mathbb{F}} \max_{\alpha_i>0} \; f_0(x) - \sum_i \alpha_i f_i(x) \; = \; \min_{x\in\mathbb{F}} \max_{\alpha_i>0} \; L(x,\alpha)$$

The function $L(x,\alpha) = f_0(x) - \sum_i \alpha_i f_i(x)$ is called the **Lagrangian**.

The **Dual Problem** switches the min-max:

$$\lambda^* = \max_{\alpha_i>0} \min_{x\in\mathbb{F}} \; f_0(x) - \sum_i \alpha_i f_i(x)$$

Or if we define $g(\alpha) = \min_{x\in\mathbb{F}} f_0(x) - \sum_i \alpha_i f_i(x)$,

$$\lambda^* = \max_{\alpha_i>0} \; g(\alpha)$$

For convex optimization, $\lambda^* = \gamma^*$. However, $x^* \neq \alpha^*$.

Note: We always have $\max_x \min_y g(x,y) \leq \min_y \max_x g(x,y)$ (2-player game).

Lagrangian Duality

$$\min_{x \in \mathbb{F}} f_0(x): \qquad \text{subject to } f_i(x) \geq 0 \qquad i = 1, \cdots k$$

Note that

$$\max_{\alpha > 0} -\alpha f_i(x) = \begin{cases} \infty & f_i(x) < 0 \\ 0 & \text{otherwise} \end{cases}$$

Equivalent Form:

$$\gamma^* = \min_{x \in \mathbb{F}} \max_{\alpha_i > 0} \; f_0(x) - \sum_i \alpha_i f_i(x) \; = \; \min_{x \in \mathbb{F}} \max_{\alpha_i > 0} \; L(x, \alpha)$$

The function $L(x, \alpha) = f_0(x) - \sum_i \alpha_i f_i(x)$ is called the **Lagrangian**.

The **Dual Problem** switches the min-max:

$$\lambda^* = \max_{\alpha_i > 0} \min_{x \in \mathbb{F}} \; f_0(x) - \sum_i \alpha_i f_i(x)$$

Or if we define $g(\alpha) = \min_{x \in \mathbb{F}} f_0(x) - \sum_i \alpha_i f_i(x)$,

$$\lambda^* = \max_{\alpha_i > 0} \; g(\alpha)$$

For convex optimization, $\lambda^* = \gamma^*$. However, $x^* \neq \alpha^*$.
Note: We always have $\max_x \min_y g(x,y) \leq \min_y \max_x g(x,y)$ (2-player game).

The property (Weak Duality)

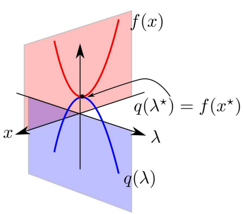
$$\max_x \min_y g(x, y) \leq \min_y \max_x g(x, y)$$

always holds for smooth functions and when equality holds can be interpreted as a Nash equilibrium (See window drawings in "A Beautiful Mind").

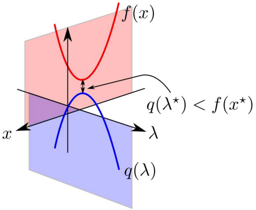
The player which moves second always wins. Note the second player here is the inner optimization problem ($\min$ on LHS and $\max$ on RHS). To verify, just use the function

$$g(x, y) = xy$$

Strong vs. Weak Duality (In the Lagrangian Sense)



strong duality



weak duality

Primal Problem:

$$\gamma^* = \min_{x \in \mathbb{R}} \quad f(x) : \qquad x \in S$$

Dual Problem:

$$\lambda^* = \max_{y \in \mathbb{R}} \quad f_D(y) : \qquad y \in S_D$$

Definition 8.

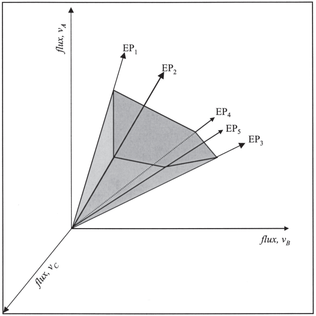
Strong Duality holds if $\lambda^* = \gamma^*$. **Weak Duality** holds if $\lambda^* < \gamma^*$.

- Strong Duality holds if f, S are convex and S has non-empty interior.
- Weak Duality always holds.
- The Lagrangian Dual Problem is **ALWAYS** Convex.

Convex Cones: What Does Positivity Even Mean?

Question: What does $f(x) \geq 0$ mean.

- What does $y \geq 0$ mean?



Definition 9.

A set is a **cone** if for any $x \in Q$,

$$\{\mu x \,:\, \mu \geq 0\} \subset Q.$$

The cone, Q, is **pointed** if $0 \in Q$.

Examples:

- **Positive Orthant:** $y \geq 0$ if $y_i \geq 0$ for $i = 1, \cdots, n$.
- **Half-space:** $y \geq 0$ if $\mathbf{a}^T y \geq 0$.
- **Intersection of Half-spaces:** $y \geq 0$ if $Ay \geq 0$ (i.e. $a_i^T y \geq 0$).
- **Positive Matrices:** $P \geq 0$ if $x^T P x \geq 0$ for all $x \in \mathbb{R}^n$.
- **Positive Functions:** $f \geq 0$ if $f(x) \geq 0$ for all $x \in \mathbb{R}^n$.

Generalized Inequalities and Convex Cones

So far, all inequalities have used the positive orthant cone

$$\min_{x \in \mathbb{F}} f_0(x): \qquad \text{subject to } f_i(x) \geq 0 \qquad i = 1, \cdots k$$

Question: Can we generalize duality to other types of inequality constraints?

Question: What is an inequality? What does ≥ 0 mean?

- An inequality implies a **partial ordering**:
  - $x \geq y$ if $x - y \geq 0$
- Any convex cone, C defines a partial ordering:
  - $x - y \geq 0$ if $x - y \in C$
- The ordering is only partial because $x \not\leq 0$ does not imply $x \geq 0$
  - $-x \notin C$ does not imply $x \in C$.
  - x may be indefinite.

Question: Which of the cones on the last slide define partial orderings?

Dual Cones and Generalized Duality (with inner-products)

Geometry of the dual problem is defined by an *inner product*, $\langle x, y\rangle$

Definition 10.

Two **Sets are Dual (**X **and** Y**)** if $x \in X$ implies $\langle x, y\rangle \geq 0$ for all $y \in Y$.

Interpretation: For every point $y \in Y$, the angle between y and every point in X is less than 90° (and vice-versa holds be definition).

For optimization problem:

$$\min_{x \in \mathbb{F}} \quad f(x): \qquad \text{subject to} \qquad g_i(x) \geq 0 \qquad i = 1, \cdots, K$$

If $x \geq 0 \leftrightarrow x \in C$ for cone C, then Lagrangian duality is now

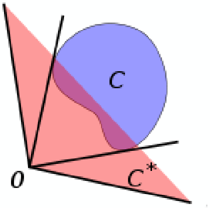
$$\min_{x \in \mathbb{F}} \; \max_{y_i \geq_* 0} \quad f(x) - \sum_{i=1}^{K} \langle y_i, g_i(x)\rangle \geq \max_{y_i \geq_* 0} \; \min_{x \in \mathbb{F}} f(x) - \sum_{i=1}^{K} \langle y_i, g_i(x)\rangle$$

where $y \geq_* 0$ means $y \in C^* := \{y \ : \ \langle y, x\rangle \geq 0 \text{ for all } x \in C\}$.

Self Dual Cones - LP and SDP

Self Dual Cones: Sometimes ≥ 0 and $\geq_*$ mean the same thing – i.e. $C = C^*$.

- **Positive Orthant:** $y \geq 0$ if $y_i \geq 0$ for $i = 1, \cdots, n$.
  - $\langle x, y \rangle = x^T y = \|x\|\|y\| \cos\theta$.
  - $x^T y \geq 0$ for all $x \geq 0$ if and only if $y \geq 0$.



- **Positive Matrices:** $P \geq 0$ if $x^T P x \geq 0$ for all $x \in \mathbb{R}^n$.
  - $\langle X, Y \rangle = \text{trace}(XY) = \sum_{ij} X_{ij} Y_{ij}$
  - $X \geq 0 \leftrightarrow X = X_{\frac{1}{2}} X_{\frac{1}{2}}^T$, so if $\langle X, Y \rangle = \text{trace}(XY) = \text{trace}(X_{\frac{1}{2}}^T Y X_{\frac{1}{2}}) \geq 0$ for any $X_{\frac{1}{2}}$, we require $Y \geq 0$.

This is why we refer to both primal and dual versions of SDP and LP.

- Their dual problems are of the same form.
- Allows Primal-Dual Algorithms
- Faster Convergence

Lagrangian Duality Examples

Two Ways to Solve the Same Problem

Primal LP:

$$\max_{x\in\mathbb{R}} \quad c^T x : \qquad Ax \le b \qquad x \ge 0$$

Dual LP:

$$\min_{y\in\mathbb{R}} \quad b^T y : \qquad A^T y \ge c \qquad y \ge 0$$

$$g(\alpha) = \max_{x\ge 0} c^T x - \alpha^T (Ax - b) = \max_{x\ge 0} (c - A^T\alpha)^T x + \alpha^T b = \begin{cases} b^T\alpha & A^T\alpha \ge c \\ \infty & \text{otherwise} \end{cases}$$

Primal SDP:

$$\min_{x\in\mathbb{R}} \quad \sum_{i=1}^m c_i x_i \qquad X = \sum_{i=1}^m F_i x_i - F_0 \ge 0$$

Dual SDP:

$$\max_{y\in\mathbb{R}} \quad \text{trace}(F_0 Y) : \qquad \text{trace}(F_i Y) = c_i \quad (i = 1, \cdots, m) \qquad Y \ge 0$$

The trace notation simply means $\text{trace}(FY) = \sum_{i,j} F_{ij} Y_{ij}$.

Lagrangian Duality Examples

Two Ways to Solve the Same Problem

Primal LP:

$$\max_{x\in\mathbb{R}} \; c^T x : \quad Ax \le b, \quad x \ge 0$$

Dual LP:

$$\min_{y\in\mathbb{R}} \; b^T y : \quad A^T y \ge c, \quad y \ge 0$$

$$g(\alpha) = \max_{x\ge 0} c^T x - \alpha^T (Ax - b) = \max_{x\ge 0} (c - A^T\alpha)^T x + \alpha^T b = \begin{cases} b^T\alpha & A^T\alpha \ge c \\ \infty & \text{otherwise} \end{cases}$$

Primal SDP:

$$\min_{x\in\mathbb{R}} \; \sum_{i=1}^{m} c_i x_i \qquad X = \sum_{i=1}^{m} F_i x_i - F_0 \ge 0$$

Dual SDP:

$$\max_{y\in\mathbb{R}} \; \text{trace}(F_0 Y) : \quad \text{trace}(F_i Y) = c_i \quad (i = 1, \cdots, m), \quad Y \ge 0$$

The trace notation simply means trace$(FY) = \sum_{i,j} F_{ij} Y_{ij}$.

$$\begin{aligned}
&\max_{x\ge 0,\, Ax-b\le 0} c^T x \\
&= \max_{x\ge 0} \; \min_{\alpha\ge 0} \; c^T x - \alpha^T (Ax - b) \\
&\le \min_{\alpha\ge 0} \; \max_{x\ge 0} \; c^T x - \alpha^T (Ax - b) \\
&= \min_{\alpha\ge 0} \; \max_{x\ge 0} \; (c^T - \alpha^T A) x + \alpha^T b \\
&= \min_{\alpha\ge 0,\, c^T - \alpha^T A \le 0} \alpha^T b \\
&= \min_{\alpha\ge 0,\, c\le A^T\alpha} b^T \alpha
\end{aligned}$$

Next Time:

More on Positive Matrices, SDP and LMIs