COMPUTATIONAL METHODS IN OPTIMIZATION

November 13, 2023

Today: Applications of SDP (More next week in the lab session)

Tomorrow: QP notebook

Next week: Lecture on (SO)CP + lab on SDP/SOCP/CP

Previously in this course

SDP (Semidefinite program)

$$U \cdot V = \mathrm{tr}(UV) = \sum_{i=1}^{n}\sum_{j=1}^{n} U_{ij}V_{ij}$$

$$(P) \quad \underset{X \in \mathbb{S}^{n\times n}}{\text{minimize}} \quad C \cdot X \quad \text{s.t.} \quad A_i \cdot X = b_i \quad i=1..m$$
$$X \succeq 0$$

symmetric matrices

$A_i \in \mathbb{S}^{n\times n}$
$C \in \mathbb{S}^{n\times n}$ $\quad b \in \mathbb{R}^m$

X positive semidefinite
$v^T X v \geq 0 \quad \forall v \in \mathbb{R}^n$

$\rightarrow$ $X \mapsto C \cdot X$
$X \mapsto A_i \cdot X - b_i$ are linear functions of X

Dual

$$\underset{\substack{y \in \mathbb{R}^m \\ S \in \mathbb{S}^{n\times n}}}{\text{maximize}} \quad b^T y \quad \text{s.t.} \quad \sum_{i=1}^{m} y_i A_i + S = C \quad \Big) \; n \times n = n^2 \text{ constraints}$$
$$S \succeq 0$$

$\hookrightarrow$ IPM for SDP :

- Consider primal-dual strictly feasible points (X^k, y^k, S^k) with $X^k \succ 0$ and $S^k \succ 0$

$U \in \mathbb{S}^{n\times n}$ $\quad U \succ 0 \iff v^T U v > 0$ when $v \neq 0_{\mathbb{R}^n}$

- Solve a Newton system to obtain a direction $(\Delta X^k, \Delta y^k, \Delta S^k)$ and compute $(X^{k+1}, y^{k+1}, S^{k+1})$ from that direction such that $(X^{k+1}, y^{k+1}, S^{k+1})$ is primal-dual strictly feasible point

$\longrightarrow$ Same theoretical guarantees for SDPs than LPs

$\longrightarrow$ But implementation (in particular solving Newton system) is very challenging

Two successful applications of SDPs

① Graph theory and Max-Cut problem

Setup: Graph (V, E)

V: vertices $\{1, \dots, n\}$

$E \subseteq \{1, \dots, n\} \times \{1, \dots, n\}$

weighted edges

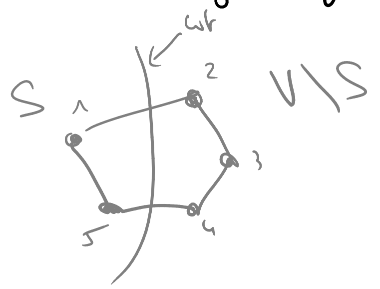
$\forall e = (i,j) \in E$, $w_{ij} \geq 0$ weight of edge (i,j)

Max-Cut problem

Find a partition of V as $S \cup (V \setminus S)$ such that

$$\delta(S) = \{(i,j) \in E : |\{i,j\} \cap S| = 1\}$$

has maximum edge weight



cut weight

$w_{12} + w_{45}$

→ the problem can be formulated as a problem in binary variables (we will use $\{-1,1\}$ variables)

$$\underset{x \in \mathbb{R}^n}{\text{maximize}} \quad x^T L x \quad \text{s.t.} \quad x_i \in \{-1, 1\} \quad \forall i = 1..n$$

Vector of partitioning for the vertices

$x_i = -1 \Rightarrow i \in S$

$x_i = +1 \Rightarrow i \in V \setminus S$

where L is the graph Laplacian matrix

$$L_{ij} = \begin{cases} -w_{ij} & \text{if } (i,j) \in E \\ 0 & \text{if } (i,j) \notin E \text{ and } i \neq j \\ \sum_{k : (i,k) \in E} w_{ik} & \text{if } i = j \end{cases}$$

↳ The problem has a quadratic objective but the constraints are integer constraints $\Rightarrow$ Discrete optimization problem

↳ We can however reformulate the problem as a <u>matrix</u> optimization problem

Key: $\forall x \in \mathbb{R}^n$, $\quad x^T L x = L \bullet \underbrace{xx^T}_{\in \mathbb{S}^{n\times n}}$

quadratic function in x

Linear function in xx^T

→ We use a matrix variable $X \in \mathbb{S}^{n\times n}$ in lieu of xx^T

→ To guarantee that X has the desired structure xx^T, we add the constraints

$$X \succeq 0 \quad (1)$$

linear constraint in X → $X_{ii} = 1 \quad \forall i = 1..n \qquad X_{ii} = x_i^2 = 1$

$$\operatorname{rank}(X) = 1 \quad (2)$$

$E_{ii} \bullet X = 1$

$$E_{ii} = \begin{array}{cc} & j \\ \begin{bmatrix} 0 & & & \\ & \ddots & & 0 \\ & & 1 & \\ 0 & & & \ddots \\ & & & & 0 \end{bmatrix} & i \end{array}$$

(1) + (2) guarantee that X can be written as xx^T for some $x \in \mathbb{R}^n$

Reformulation

$$\underset{X \in \mathbb{S}^{n\times n}}{\text{maximize}} \quad L \bullet X \quad \text{s.t.} \quad \begin{array}{l} X_{ii} = 1 \quad \forall i = 1..n \\ X \succeq 0 \\ \operatorname{rank}(X) = 1 \end{array}$$

Almost an SDP: rank constraint is the only difference between this problem and an SDP

$\Rightarrow$ Rank constraint hard to deal with!

$\Rightarrow$ To get an SDP, we drop the rank constraint ("we relax the constraint")

SDP relaxation

$$\underset{X \in \mathbb{S}^{n\times n}}{\text{minimize}} \quad (-L) \cdot X \quad \text{s.t.} \quad X_{ii} = 1, \quad X \succeq 0$$

Max-cut SDP

→ The relaxation is not equivalent to the original problem in general
→ The solution of the relaxation can be used to obtain a good approximation to the solution of the original problem
(Goemans & Williamson)

1. Solve Max-Cut SDP ⇒ $X^* \in \mathbb{S}^{n\times n}$ $\quad X^* \succeq 0 \quad X^*_{ii} = 1$
2. Write X^* as $[v_i^T v_j]_{\substack{1\le i\le n \\ 1\le j\le n}}$ for some vectors $v_1, -, v_n \in \mathbb{R}^n$, $\|v_i\| = 1 \ \forall i$

   $X^*_{ij} = v_i^T v_j$

   ⇒ Always possible using linear algebra techniques
   ⇒ Use n vectors to rewrite X^*
3. Draw a random vector in the unit sphere $u \in \{x \in \mathbb{R}^n \,/\, \|x\| = 1\}$ (uniformly) and define $x^* \in \mathbb{R}^n$

$$\forall i = 1..n, \quad x_i^* = \begin{cases} -1 & \text{if } u^T v_i \le 0 \\ 1 & \text{if } u^T v_i > 0 \end{cases}$$

⇒ Defines a "random cut" based on X^*

Th) If $x_{opt} \in \mathbb{R}^n$ is the solution of the Max-Cut problem then

$$\underbrace{x_{opt}^T L x_{opt} \geq (x^*)^T L (x^*)}_{x^* \text{ feasible}} \underbrace{\geq 0.87856\, x_{opt}^T L x_{opt}}_{x^* \text{ built from } X^*}$$

$\Rightarrow x^*$ is a 0.87856-approximation of x_{opt}

→ SDPs are often used to define relaxations (i.e. approximations) of graph problems

→ More generally, SDPs are a useful tool to reformulate problems on binary variables with quadratic objectives

2 Matrix completion

<u>Problem</u>: Data matrix $M \in \mathbb{R}^{m \times n}$

Observe a <u>subset</u> of the entries of M (e.g. Netflix movie/user preferences)

<u>Goal</u>: Compute $W \in \mathbb{R}^{m \times n}$ that approximates M and that has the <u>lowest rank</u> possible

Simplest approximation possible

<u>Basic optimization formulation</u>

$$\underset{W \in \mathbb{R}^{m \times n}}{\text{minimize}} \quad \text{rank}(W) \quad \text{s.t.} \quad W_{ij} = M_{ij} \quad \forall (i,j) \text{ observed entry of } M$$

→ The rank function is hard to deal with

→ We can add more variables to the problem and turn it into an SDP.

$$\underset{\substack{W \in \mathbb{R}^{m \times n} \\ Y \in \mathbb{S}^{m \times m} \\ Z \in \mathbb{S}^{n \times n}}}{\text{minimize}} \quad \text{trace}(Y) + \text{trace}(Z) \quad \text{s.t.} \quad W_{ij} = M_{ij} \quad \forall (ij) \text{ observed}$$

$$\begin{bmatrix} Y & W \\ W^T & Z \end{bmatrix} \succeq 0$$

(from Candès & Recht 2009)

↳ Set $X = \begin{bmatrix} Y & W \\ W^T & Z \end{bmatrix} \in \mathbb{S}^{(m+n)\times(m+n)}$. The problem is then a SDP in X

$$\operatorname{trace}(Y) + \operatorname{trace}(Z) = I \bullet X = \operatorname{trace}(IX) = \operatorname{trace}(X)$$

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{(m+n)\times(m+n)}$$

$$W_{ij} = M_{ij} \iff E_{i(m+j)} \bullet X = M_{ij}$$

$$E_{i(m+j)} = \begin{array}{c|c} & m+j \\ \hline i & \begin{bmatrix} \cdots & 0 & \vdots & 0 \\ \cdots & \cdots & 1 & \cdots \\ & & 0 & \end{bmatrix} \end{array} \updownarrow m+n$$

$$\overleftrightarrow{\;m+n\;}$$

$$\iff E_{(m+j)i} \bullet X = M_{ij}$$

$$\iff \left(\frac{E_{i(m+j)} + E_{(m+j)i}}{2}\right) \bullet X = M_{ij}$$

(a) $\underset{X \in \mathbb{S}^{n\times n}}{\text{minimize}} \quad C \bullet X \quad \text{s.t.} \quad A_i \bullet X = b_i$

$$X \succeq 0$$

(b) $\underset{X \in \mathbb{R}^{n\times n}}{\text{minimize}} \quad C \bullet X \quad \text{s.t.} \quad A_i \bullet X = b_i$

$$\left. X_{ij} = X_{ji} \quad \forall i \neq j \right\} \frac{n^2 - n}{2} \text{ constraints}$$

$$X \succeq 0$$

(b) $\Rightarrow$ Deal with n^2 coefficients + symmetry constraints

(a) $\Rightarrow$ Can reduce the problem to $\frac{n^2+n}{2}$ coefficients + benefit from the operations defined on $\mathbb{S}^{n\times n}$