Approximation Algorithms and Quadratic Form Maximization over Convex Sets

Vijay Bhattiprolu

Intro to Approximation Algorithms

Approximation Algorithms and Hardness of Approximation:

Intro to Approximation Algorithms

Approximation Algorithms and Hardness of Approximation:

- Many optimization tasks of interest are believed to be impossible to solve exactly (by polytime algorithms) but can be solved approximately.

Intro to Approximation Algorithms

Approximation Algorithms and Hardness of Approximation:

- Many optimization tasks of interest are believed to be impossible to solve exactly (by polytime algorithms) but can be solved approximately.
- Which optimization problems admit polynomial time algorithms computing a solution optimal (multiplicatively) within an absolute constant?

Intro to Approximation Algorithms

Approximation Algorithms and Hardness of Approximation:

- Many optimization tasks of interest are believed to be impossible to solve exactly (by polytime algorithms) but can be solved approximately.
- Which optimization problems admit polynomial time algorithms computing a solution optimal (multiplicatively) within an absolute constant?
- What do such algorithms typically look like? (Today: Convex Programming)

Intro to Approximation Algorithms

Approximation Algorithms and Hardness of Approximation:

- Many optimization tasks of interest are believed to be impossible to solve exactly (by polytime algorithms) but can be solved approximately.
- Which optimization problems admit polynomial time algorithms computing a solution optimal (multiplicatively) within an absolute constant?
- What do such algorithms typically look like? (Today: Convex Programming)
- Can one prove a certain algorithm achieves the optimal constant? (Assuming $\mathrm{P} \neq \mathrm{NP}$ or similar hypotheses) (related to Probabilistically Checkable Proofs)

Example: Max-Cut in a Graph

MAX-CUT: Given a graph $G = (V, E)$ as input, partition the vertices so that the maximum number of edges cross the partition.

Example: Max-Cut in a Graph

MAX-CUT: Given a graph $G = (V, E)$ as input, partition the vertices so that the maximum number of edges cross the partition.

Simple 2-approximation algorithm: partitioning randomly will cut at least $(1 - o(1))|E|/2$ edges with high probability.

Example: Max-Cut in a Graph

MAX-CUT: Given a graph $G = (V, E)$ as input, partition the vertices so that the maximum number of edges cross the partition.

Simple 2-approximation algorithm: partitioning randomly will cut at least $(1 - o(1))|E|/2$ edges with high probability.

~ 1.14-approximation is possible using Convex Programming!

Example: Max-Cut in a Graph

MAX-CUT: Given a graph $G = (V, E)$ as input, partition the vertices so that the maximum number of edges cross the partition.

Simple 2-approximation algorithm: partitioning randomly will cut at least $(1 - o(1))|E|/2$ edges with high probability.

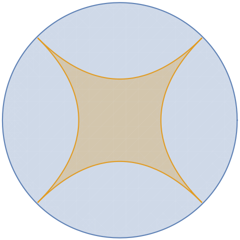
~ 1.14-approximation is possible using Convex Programming!

This Convex Programming algorithm achieves the optimal constant assuming the Unique Games Conjecture [Khot, Kindler, Mossel , O'Donnell, Oleszkiewicz]

Convex Relaxation + Rounding Paradigm

Given combinatorial optimization problem.

Convex Relaxation + Rounding Paradigm



Given combinatorial optimization problem.

(1) "Relax" problem to convex program.

Interlude: Convex Relaxation

$$\sup_{x \in S_1} \langle a, x \rangle \leq \sup_{x \in S_2} \langle a, x \rangle$$

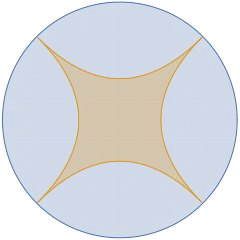
Relax the complicated set S_1 to a larger convex set S_2 (with a membership oracle).

Convex Relaxation + Rounding Paradigm

Given combinatorial optimization problem.

(1) "Relax" problem to convex program.

(2) Compute the exactly optimal solution to the relaxation.



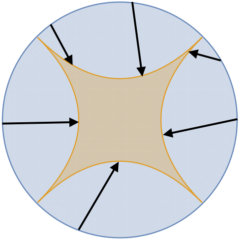
Convex Relaxation + Rounding Paradigm

Given combinatorial optimization problem.

(1) "Relax" problem to convex program.

(2) Compute the exactly optimal solution to the relaxation.

(3) Map solution back to original region (Rounding Algorithm)



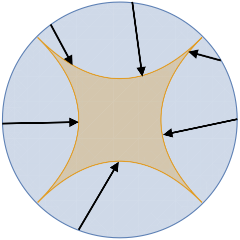
Convex Relaxation + Rounding Paradigm

Given combinatorial optimization problem.

(1) "Relax" problem to convex program.

(2) Compute the exactly optimal solution to the relaxation.

(3) Map solution back to original region (Rounding Algorithm)



Often the best polytime approximation algorithm. (Under complexity assumptions.)

Convex Vector Relaxation for MAX-CUT

MAX-CUT reformulation:

$$\mathrm{OPT} = \ \mathsf{sup} \ \sum_{ij \in E} (1 - x_i x_j)/2 \quad \text{s.t. } \forall i, \ x_i \in \{\pm 1\}$$

Natural Vector Relaxation:

$$\mathrm{CP} = \ \mathsf{sup} \ \sum_{ij \in E} (1 - \langle u_i, u_j \rangle)/2 \ \text{ s.t. } \forall i, \ \|u_i\|_2 = 1,$$

Convex Vector Relaxation for MAX-CUT

MAX-CUT reformulation:

$$\mathrm{OPT} = \ \mathsf{sup} \ \sum_{ij \in E} (1 - x_i x_j)/2 \quad \text{s.t. } \forall i, \ x_i \in \{\pm 1\}$$

Natural Vector Relaxation:

$$\mathrm{CP} = \ \mathsf{sup} \ \sum_{ij \in E} (1 - \langle u_i, u_j \rangle)/2 \ \text{ s.t. } \forall i, \ \langle u_i, u_i \rangle = 1,$$

Convex Vector Relaxation for MAX-CUT

MAX-CUT reformulation:

$$\mathrm{OPT} = \ \mathsf{sup} \ \sum_{ij \in E} (1 - x_i x_j)/2 \quad \text{s.t. } \forall i,\ x_i \in \{\pm 1\}$$

Natural Vector Relaxation:

$$\begin{aligned} \mathrm{CP} &= \ \mathsf{sup} \ \sum_{ij \in E} (1 - \langle u_i,\ u_j \rangle)/2 \ \ \text{s.t. } \forall i,\ \langle u_i, u_i \rangle = 1, \\ &= \ \mathsf{sup} \ \ \langle D - A, \mathbb{X} \rangle /2 \ \ \text{s.t.} \quad \mathbb{X} \succeq 0, \quad \forall i,\ \ \mathbb{X}_{i,i} = 1 \end{aligned}$$

(Substituting $\mathbb{X}_{i,j} := \langle u_i,\ u_j \rangle$)

Convex Vector Relaxation for MAX-CUT

MAX-CUT reformulation:

$$\mathrm{OPT} = \sup \sum_{ij \in E} (1 - x_i x_j)/2 \quad \text{s.t. } \forall i,\ x_i \in \{\pm 1\}$$

Natural Vector Relaxation:

$$\begin{aligned} \mathrm{CP} &= \sup \sum_{ij \in E} (1 - \langle u_i,\ u_j \rangle)/2 \ \text{ s.t. } \forall i,\ \langle u_i, u_i \rangle = 1, \\ &= \sup \ \langle D - A, \mathbb{X} \rangle / 2 \ \text{ s.t. } \ \mathbb{X} \succeq 0,\ \ \forall i,\ \mathbb{X}_{i,i} = 1 \\ &\quad (\text{Substituting } \mathbb{X}_{i,j} := \langle u_i,\ u_j \rangle) \end{aligned}$$

Rounding Algorithm: Choose a random hyperplane through the origin and partition vectors according to it.

Convex Vector Relaxation for MAX-CUT

MAX-CUT reformulation:

$$\mathrm{OPT} = \sup \sum_{ij \in E} (1 - x_i x_j)/2 \quad \text{s.t. } \forall i,\ x_i \in \{\pm 1\}$$

Natural Vector Relaxation:

$$\begin{aligned}\mathrm{CP} &= \sup \sum_{ij \in E} (1 - \langle u_i,\, u_j \rangle)/2 \ \text{ s.t. } \forall i,\ \langle u_i, u_i \rangle = 1,\\ &= \sup \ \langle D - A, \mathbb{X} \rangle /2 \ \text{ s.t. } \ \mathbb{X} \succeq 0,\ \ \forall i,\ \mathbb{X}_{i,i} = 1\end{aligned}$$

$$(\text{Substituting } \mathbb{X}_{i,j} := \langle u_i,\, u_j \rangle)$$

Rounding Algorithm: Choose a random hyperplane through the origin and partition vectors according to it.
[Goemans Williamson 97]: Achieves ~ 1.14 approximation

Optimality of Relaxation + Rounding Paradigm

[Raghavendra 08]

- Given a Constraint Satisfaction Problem.
- A natural Convex Programming relaxation is the best polytime apx. alg. under Khot's Unique Games Conjecture.

My Interests: Quadratic Maximization

Goal: Polynomial time Approximation Algorithm

Input: $A \in \mathbb{R}^{n \times n}$ and an oracle computing the norm $(\| \cdot \|_X, \mathbb{R}^n)$.

Compute in polynomial time (approximately):

$$\sup_{\|x\|_X \leq 1} \langle x, Ax \rangle = \sum_{i,j} A_{i,j} \cdot x_i \cdot x_j \qquad \text{Quadratic Maximization}$$

My Interests: Quadratic Maximization

Goal: Polynomial time Approximation Algorithm

Input: $A \in \mathbb{R}^{n \times n}$ and an oracle computing the norm $(\|\cdot\|_X, \mathbb{R}^n)$.

Compute in polynomial time (approximately):

$$\sup_{\|x\|_X \leq 1} \langle x, Ax \rangle = \sum_{i,j} A_{i,j} \cdot x_i \cdot x_j \qquad \text{Quadratic Maximization}$$

Very rich class. Captures tractable and highly intractable problems

Examples of Quadratic Maximization

- ℓ_2: Maximum Eigenvalue. Exactly computable.

Examples of Quadratic Maximization

- ℓ_2: Maximum Eigenvalue. Exactly computable.
- ℓ_∞ for Laplacian A: MAX-CUT in a Graph.
  ~ 1.14 Apx. [Goemans Williamson 97]

Examples of Quadratic Maximization

- ℓ_2: Maximum Eigenvalue. Exactly computable.
- ℓ_∞ for Laplacian A: MAX-CUT in a Graph.
  ~ 1.14 Apx. [Goemans Williamson 97]
- ℓ_∞: Grothendieck's Inequality. $O(1)$ Apx.
  [Grothendieck 53] . . . [Braverman (Makarychev)^2 Naor 11].

Examples of Quadratic Maximization

- ℓ_2: Maximum Eigenvalue. Exactly computable.
- ℓ_∞ for Laplacian A: MAX-CUT in a Graph.
  ~ 1.14 Apx. [Goemans Williamson 97]
- ℓ_∞: Grothendieck's Inequality. $O(1)$ Apx.
  [Grothendieck 53] … [Braverman (Makarychev)^2 Naor 11].
- ℓ_p for $p < 2$: Related to Hypercontractivity. $n^{\Omega(1)}$ is best Apx.

Examples of Quadratic Maximization

- ℓ_2: Maximum Eigenvalue. Exactly computable.
- ℓ_∞ for Laplacian A: MAX-CUT in a Graph.
  ~ 1.14 Apx. [Goemans Williamson 97]
- ℓ_∞: Grothendieck's Inequality. $O(1)$ Apx.
  [Grothendieck 53] . . . [Braverman (Makarychev)^2 Naor 11].
- ℓ_p for $p < 2$: Related to Hypercontractivity. $n^{\Omega(1)}$ is best Apx.
- (Schatten) S_∞: $O(1)$ Apx.
  Non-commutative Grothendieck Inequality, Quantum Information Theory, etc.
  [Pisier 78, Haagerup 87] [Naor Regev Vidick 12].

Questions of Interest

Goal: Extend Theory of Approximation Algorithms to Quadratic Maximization.

Questions of Interest

Goal: Extend Theory of Approximation Algorithms to Quadratic Maximization.

Hope: Theory can be used for both Algorithmic and Impossibility results for Continuous/Combinatorial Optimization.

Questions of Interest

- How does Approximability depend on Geometry of X? When do $O(1)$ Approximation Algorithms exist?

Questions of Interest

- How does Approximability depend on Geometry of X? When do $O(1)$ Approximation Algorithms exist?
- When is Convex Programming the Optimal Algorithm?

Recent Joint work with Euiwoong Lee and Assaf Naor

Generic Framework for Quadratic Maximization:

Recent Joint work with Euiwoong Lee and Assaf Naor

Generic Framework for Quadratic Maximization:

- Encompasses situations where $O(1)$-approximation algorithms were known.

Recent Joint work with Euiwoong Lee and Assaf Naor

Generic Framework for Quadratic Maximization:

- Encompasses situations where $O(1)$-approximation algorithms were known.
- A rich family of new examples where $O(1)$-approximation is possible.

Recent Joint work with Euiwoong Lee and Assaf Naor

Generic Framework for Quadratic Maximization:

- Encompasses situations where $O(1)$-approximation algorithms were known.
- A rich family of new examples where $O(1)$-approximation is possible.
- Characterization (under complexity assumptions) for special families:
  (a) Norms invariant to permutations and sign-flips
  (b) Unitarily Invariant Matrix Norms.

Thank You. Questions?