Rounding Techniques in Approximation Algorithms

Lecture 21: Introduction to Sum of Squares

Lecturer: Nathan Klein

1 Preamble: LP Duality

So far, we have focused on directly rounding optimal solutions to linear programs without looking at the role of the dual. Every LP has a dual program that we can view as a simple proof system. Consider the following LP modeling the max independent set problem on a triangle graph:

$$\begin{aligned} \max \quad & x_u + x_v + x_w \\ \text{s.t.} \quad & x_u + x_v \leq 1 \\ & x_u + x_w \leq 1 \\ & x_v + x_w \leq 1 \\ & 0 \leq x_u, x_v, x_w \end{aligned}$$

The dual is now allowed to use the constraints as proof lines, and the goal is to prove that $x_u + x_v + x_w \leq \alpha$ for the smallest α possible. The dual is only allowed to multiply constraints by real numbers and add them together.

In this case, we can see a primal solution $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$ is feasible. Can we prove it's optimal? Yes: the dual can take $\frac{1}{2}$ times each of the first three constraints and add them together, yielding $x_u + x_v + x_w \leq \frac{3}{2}$.

Of course, the true answer is 1 here. Can we somehow strengthen our proof system to show this? One method of doing so is the Sherali-Adams proof system [SA90], which I encourage you to explore. However, over the next few lectures we will focus on a stronger system known as Sum-of-Squares. In this lecture we will give the basics of this proof system.

2 Non-negativity of Polynomials over the Cube

Consider the following problem. Given a polynomial over $x_1, \ldots, x_n$ with rational coefficients, determine if $p(x) \geq 0$ for all $x \in \{-1, 1\}^n$ or find a point $x \in \{-1, 1\}^n$ for which $p(x) < 0$. In other words, determine if p is non-negative over the hypercube.

It sounds relatively harmless as first, but it turns out to be NP-Hard. For example, an algorithm for this problem can solve max cut. Define

$$p_M = \frac{1}{2} \sum_{\{u,v\} \in E} (1 - x_u x_v)$$

as the max cut polynomial for a graph $G = (V, E)$, where we make a variable x_v for all $v \in V$. It's called the max cut polynomial because for $x \in \{-1, 1\}^V$, $p_M(x)$ is exactly the number of edges in the cut when you put all vertices v with $x_v = 1$ on one side and all other vertices on the other side. Now, the polynomial $\alpha - p_M$ is non-negative over the hypercube if and only if $p_M(x) \leq \alpha$

for all $x \in \{-1,1\}^V$. But this is equivalent to certifying that the value of the max cut is at most α, which is an NP-Hard task.

Let's consider the following simple example of p_M for the triangle graph on vertices u, v, w, which we know has a max cut of value at most 2. Thus, $2 - p_M$ is non-negative. How might we realize this?

$$2 - p_M = 2 - \frac{1}{2}(1 - x_u x_v + 1 - x_u x_w + 1 - x_v x_w) = \frac{1}{2}(1 + x_u x_v + x_u x_w + x_v x_w)$$

Now, where we use crucially that $x_i^2 = 1$ for all x in the cube:

$$0 \leq (1 + x_u x_v + x_u x_w + x_v x_w)^2 = 4 + 2x_u x_v + 2x_u x_w + 2x_v x_w$$

And this implies that $2 - p_M$ is also non-negative over the cube since it is just a scaling of this polynomial. Before we move on, I want to highlight two things:

1. $2 - p_M$ is **not** non-negative over $\mathbb{R}$. We needed to use that $x_i^2 = 1$ over the cube. For example, we could plug in $x_u = 0$, $x_v = 1$, $x_w = -100$ to obtain a negative value.

2. In this example, we obtain **a better lower bound for the triangle than the max cut SDP.** Recall that there, the SDP could only certify that the value of the max cut was at most $\frac{9}{4}$, which is a worse lower bound.

This second point suggests that something interesting might be going on here. We will dive into this in detail over the next few lectures.

2.1 Sums of Squares

The technique we used to prove that $2 - p_M$ was non-negative was to write its value on the cube as a sum of squares of polynomials. It turns out this is a powerful, very general proof technique.

Definition 2.1 (Degree d Sum-of-Squares (SoS) Certificate)**.** *We say a polynomial p of maximum degree d has a degree d SoS certificate of non-negativity (over the cube) if there exists polynomials $g_1, \ldots, g_r$, $\deg(g_i) \leq \frac{d}{2}$, so that*

$$p(x) = \sum_{i=1}^{r} g_i^2 \quad \forall x \in \{-1,1\}^n$$

It turns out if p has a degree d SoS certificate, then for any $\epsilon > 0$ we can find a degree d SoS certificate for $p + \epsilon$ in time $\text{poly}(n^d, \log \frac{1}{\epsilon})$. This can be proved using the ellipsoid method or an SDP, as we will see later. In this course, we don't care about losing an ϵ, which can be made as small as 2^{-n}, so usually we will sweep this under the rug. For now, let's show that certificates can be efficiently checked (when d is a constant) and that every non-negative polynomial has a certificate.

Lemma 2.2. *Given a certificate, we can check if it is correct in time polynomial in n^d.*

Proof. We can write p using its unique representation $p = \sum \hat{p}(S) \prod_{i \in S} x_i$ (formally, we would say $\hat{p}(S)$ is the Fourier coefficient of $x^S = \prod_{i \in S} x_i$). To see this all you need to do is multiply p out and repeatedly replace squared terms with 1 (as recall they will be 1 over the cube), or, somewhat

more efficiently, replace x_i^k for $k \geq 2$ with $x_i^{k \pmod 2}$. Since p has maximum degree d, there are at most n^d terms, so this is an efficient process.

But now, do the same thing for the SoS certificate $\sum g_i^2$, which again has degree at most d. Now just check if all the coefficients are the same. If they aren't, it's not a certificate by the uniqueness of the representation $\sum \hat{p}(S) \prod_{i \in S} x_i$. □

This also shows that SoS certificates have polynomial size so long as d is a constant. So this is a reasonable framework so far for constant d. Another nice fact is as follows:

Lemma 2.3. *If p is non-negative over the cube, then there is a certificate of degree $2n$.*

Proof. Let $g(x) = \sqrt{p(x)}$. A standard fact is that every function over a finite field is a polynomial[1]. But, over the cube the polynomial for g is multilinear and therefore has degree at most n, giving the proof. □

2.2 Tensors

Notice that for the triangle max cut polynomial $2 - p_M$, we gave a degree 4 proof of non-negativity. It turns out that the (dual of the) SDP we solved for max cut can only find degree 2 proofs of non-negativity. This explains why we obtained a better lower bound in this case. A major open question is whether higher (but still constant) degree SoS can lead to a better approximation algorithm for max cut. We will use tensor notation to better understand why our previous SDP was finding degree 2 certificates and show how we can find degree d SoS certificates in time polynomial in n^d.

Given $v \in \mathbb{R}^n$, $v^{\otimes k} \in \mathbb{R}^{n^k}$, is the kth tensor power of v. We will index these n^k coordinates by writing $v^{\otimes k}(i_1, i_2, \ldots, i_k)$ for $i_j \in [n]$ and define

$$v^{\otimes k}(i_1, i_2, \ldots, i_k) = \prod_{j=1}^{k} v_{i_j}$$

Using this, we can prove the following, where $(1, x)$ indicates the vector x with a 1 appended in the first coordinate. So, $(1, x)^{\otimes d/2}$ indicates a vector with entries corresponding to every monomial of degree at most $d/2$.

Lemma 2.4. *A polynomial p over the cube has a degree d SoS certificate if and only if there is a matrix $A \in \mathbb{R}^{(n+1)^{d/2} \times (n+1)^{d/2}}$ so that $A \succeq 0$ and $p(x) = ((1, x)^{\otimes d/2})^T A (1, x)^{\otimes d/2}$ on all $x \in \{-1, 1\}^n$.*

Proof. First, suppose such a matrix A exists. Then, $A = BB^T$ for some B, and for all x in the cube,

$$p(x) = ((1, x)^{\otimes d/2})^T BB^T (1, x)^{\otimes d/2} = \|B^T (1, x)^{\otimes d/2}\|_2^2$$

But the entries of the vector $B^T (1, x)^{\otimes d/2}$ are polynomials of degree at most $d/2$ (as they are linear combinations of monomials of degree at most $d/2$) so this is an SoS certificate.

For the other direction, suppose p has a degree d SoS certificate. Then, for all x in the cube,

$$p(x) = \sum_{i=1}^{r} g_i^2$$

[1] An easy way to see this over the cube is to create a monomial encoding the value of each possible x which is 0 elsewhere.

where each g_i has maximum degree $d/2$. Let's construct our matrix $A \succeq 0$. Notice that for each i, we have $g_i(x) = v_i^T(1,x)^{\otimes d/2}$ for some vector $v_i \in \mathbb{R}^{n^{d/2}}$ (put the Fourier coefficient of each monomial in v_i). So,

$$p(x) = \sum_{i=1}^{d}((1,x)^{\otimes d/2})^T v_i v_i^T (1,x)^{\otimes d/2} = ((1,x)^{\otimes d/2})^T \left(\sum_{i=1}^{d} v_i v_i^T\right)(1,x)^{\otimes d/2}$$

So, we can define a matrix B with the v_i vectors as its columns and $A = BB^T$. □

References

[SA90] Hanif D. Sherali and Warren P. Adams. "A Hierarchy of Relaxations between the Continuous and Convex Hull Representations for Zero-One Programming Problems". In: *SIAM Journal on Discrete Mathematics* 3.3 (1990), pp. 411–430. DOI: 10.1137/0403036. eprint: https://doi.org/10.1137/0403036 (cit. on p. 1).