Information and Coding Theory **Autumn 2017**

Lecture 11: November 1, 2017

Lecturer: Madhur Tulsiani

1 Matrix Scaling

We will consider an application of I-projections to a problem known as matrix scaling. Say we are given two nonnegative matrices $M, N \in \mathbb{R}_+^{n \times n}$ such that for all i, j, $M_{ij} = 0 \Leftrightarrow N_{ij} = 0$. The goal is to multiply (scale) each row i of M by a number r_i and each column j by c_j, such that the resulting matrix M' has the same row and column sums as the target matrix N. Another way of stating this is that we want to find diagonal matrices R and C such that for $M' = RMC$, we have

$$\sum_j M'_{ij} \;=\; \sum_j N_{ij} \quad \forall i \in [n] \qquad \text{and} \qquad \sum_i M'_{ij} \;=\; \sum_i N_{ij} \quad \forall j \in [n]\,.$$

We will show a special case when the goal is to scale M so that the resulting matrix M' is doubly stochastic i.e.,

$$\sum_j M'_{ij} \;=\; \sum_i M'_{ij} \;=\; 1 \quad \forall i, j \in [n]\,.$$

First, note that by a *global* scaling of $1/\sum_{i,j} M_{ij}$, we can assume that $\sum_{i,j} M_{ij} = 1$ and the goal is instead to scale it to have row and column sums equal to $1/n$. We assume that $M_{ij} > 0$ for all i, j (we can think of the target matrix N with $N_{ij} = 1/n^2$ for all i, j). Since the entries of M are positive and sum to 1, we can think of it as a probability distribution Q with $\mathrm{Supp}(Q) = [n] \times [n]$ (where $q(i, j) = M_{ij}$). We consider the linear family of distributions on $[n] \times [n]$ (written as matrices) with the required row and column sums.

$$\mathcal{L} \;:=\; \left\{ P \;\mid\; \sum_j p(i,j) \;=\; \sum_i p(i,j) \;=\; \frac{1}{n} \quad \forall i, j \in [n] \right\}$$

Note that the abobe is a linear family as defined in the previous lecture, since we can consider functions $f_1, \ldots, f_n$ and $g_1, \ldots, g_n$ defined as

$$f_{i_0}(i,j) \;=\; \begin{cases} 1 & \text{if } i = i_0 \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad g_{j_0}(i,j) \;=\; \begin{cases} 1 & \text{if } j = j_0 \\ 0 & \text{otherwise} \end{cases}.$$

Then, the above family can be written in terms of the expectations of the functions f_i and g_j for all $i, j \in [n]$. Moreover, we know from the previous lecture that the I-projection P^* of Q onto $\mathcal{L}$ is of the form

$$p^*(i,j) \;=\; c_0 \cdot q(i,j) \cdot \exp\left(\sum_{i_0} \lambda_{i_0} \cdot f_{i_0}(i,j) + \sum_{j_0} \mu_{j_0} \cdot g_{j_0}(i,j)\right) \;=\; c_0 \cdot q(i,j) \cdot \exp\left(\lambda_i + \mu_j\right) .$$

Thus, scaling each row i of M by $\sqrt{c_0} \cdot \exp(\lambda_i)$ and each column j by $\sqrt{c_0} \cdot \exp(\mu_j)$ ensures that all row and column sums are $1/n$. Combining this with the global scaling, we can also get all the row and column sums to be 1 (i.e., make the matrix doubly stochastic).

Exercise 1.1 *Where did we use the fact that $M_{ij} > 0$ for all $i, j \in [n]$?*

Exercise 1.2 *Use this the above techniques to solve the matrix scaling problem for an arbitrary target matrix N (assuming $M_{ij} = 0 \Leftrightarrow N_{ij} = 0$).*

Matrix scaling and its generalization, known as operator scaling have found a variety of applications in combinatorial optimization, complexity theory and analysis. Please take a look at the recent tutorial by Wigderson [Wig17] for an introduction to many of these connections.

2 Error-Correcting Codes

Over the next few lectures, we will switch to the "coding theory" part of the course and see how to construct (and work with) error correcting codes. We first review some basic notions about finite fields that we will need.

2.1 Some relevant facts about finite fields

We will use the fact that the set $\mathbb{F}_p = \{0, \ldots, p-1\}$ is a field for prime p, with all arithmetic operations defined modulo p. There also exist finite fields of size p^r for any prime p and positive integer r, but we will mostly restrict our discussion to prime fields. We recall some basic facts:

- $\mathbb{F}_p^n$ is a vector space over the field $\mathbb{F}_p$. Note this is *not* an inner product space.
- **Fermat's little theorem:** $a^p \equiv a \mod p$, for all $a \in \{0, \ldots, p-1\}$.
- The set of all polynomials in a single variable x, over $\mathbb{F}_p$, is defined as

$$\mathbb{F}_p[x] \;:=\; \left\{ c_0 + c_1 \cdot x + \cdots + c_{p-1} \cdot x^{p-1} \;\mid\; c_0, \ldots, c_{p-1} \in \mathbb{F}_p \right\} .$$

Note that the degree (highest power of x with a non-zero coefficient) is never more than $p-1$ (why?)

- A polynomial of degree d, which is not identically zero, has at most d roots.
- **Lagrange interpolation:** Given *distinct* $a_1, \ldots, a_{d+1} \in \mathbb{F}_p$ and any $b_1, \ldots, b_{d+1} \in \mathbb{F}_p$, the *unique* polynomial Q with degree at most d, satisfying $Q(a_i) = b_i$ for all $i \in [d+1]$, is given by

$$Q(x) \;=\; \sum_{i=1}^{d+1} b_i \cdot \prod_{j \neq i} \left(\frac{x - a_j}{a_i - a_j} \right) .$$

2.2 The Shannon and Hamming models

Suppose Alice wants to send Bob a message over a noisy channel, where some of the symbols in Alice's message may become corrupted. Let's assume that Alice's message, is a sequence of elements of $\mathbb{F}_p$ for some prime p. If the original message Alice wants to send is $m = m_1, \ldots, m_k$, she will encode with some encoding function $\mathsf{Enc} : \mathbb{F}_p^k \to \mathbb{F}_p^n$, where the goal of the encoding is to introduce some redundancy (to handle errors in transmission). Say $\mathsf{Enc}(m) = x$. Alice will transmit x and Bob will receive corrupted version y. He will then apply some decoding function $\mathsf{Dec} : \mathbb{F}_p^n \to \mathbb{F}_p^k$, to try and figure our the message. The goal is to define encoding and decoding functions such that $\mathsf{Dec}(y) = m$ even when y is a corrupted version of x. Of course, it is impossible to handle arbitrary corruptions. We will consider two models of how the corruption might occur.

Shannon model. Let $x = x_1, x_2, \ldots, x_n$ be a string where each $x_i \in \mathbb{F}_p$. In the Shannon Model, Alice sends x over the communication channel, and each x_i is corrupted independently (though not necessarily identical distributions for each bit) at random.

Example 2.1 (Binary Symmetric Channel) *Suppose the message is a tuple from $\mathbb{F}_2$. Then, for some $\varepsilon \in [0, \frac{1}{2}]$, each bit is flipped independently at random with probability ε, and is transmitted without error with probability $1 - \varepsilon$.*

We will work with a different model, which is more relevant for some of the applications in theoretical computer science.

Hamming model Suppose Alice sends n symbols $x_1, x_2, \ldots, x_n$ from $\mathbb{F}_p$. In the Hamming Model, up to t of these symbols are corrupted (but not lost) in some arbitrary manner. There is no way to know which of the symbols have been corrupted, and which symbols were transmitted correctly. We wish to design encoding and decoding schemes to recover up to t errors for some fixed t.

2.3 Codes and distances

Definition 2.2 *A code is a subset $C \subseteq \mathbb{F}_p^n$. If $|C| = p^k$, we think of the code as encoding messages in $\mathbb{F}_p^k$. The quantity k is called the* message length *and n is called the* block length *of C. We call $R(C) = \frac{k}{n}$ the* rate *of C.*

Remark 2.3 *Often the code C is implicitly associated with the encoding function $\mathsf{Enc} : \mathbb{F}_p^k \to C$. For $x \in \mathbb{F}_p^k$, we may refer to $\mathsf{Enc}(x)$ simply as $C(x)$. The use of C as a set or a function will be clear from the context.*

Our goal is to introduce some redundancies to our message such that we can uniquely recover the correct k symbols given that up to t bits from the n bit message may be corrupted.Consider the following trivial code.

Example 2.4 *Define $C : \mathbb{F}_2^4 \to \mathbb{F}_2^{12}$ as*

$$C(x_1x_2x_3x_4) = (x_1x_1x_1x_2x_2x_2x_3x_3x_3x_4x_4x_4)\,.$$

It is easy to see that the above code can handle one error. When Bob receives the string, he simply takes the majority of each block of three symbols and thus recovers the original message. As we shall see, no more than one error can be recovered using this code. Also, the rate of this code is $1/3$.

In general we'd like to keep this rate close to 1, as this would imply we introduce fewer redundancies to the encoded message. To better understand how many errors can be corrected using a particular code, we define the *distance* of a code.

Definition 2.5 *Let $C \subseteq \mathbb{F}_p^n$ be a code. Let $\delta(x, y)$ denote the Hamming distance of two strings x, y. We define the distance of a code $\Delta(C)$ as*

$$\Delta(C) \;:=\; min_{\substack{x \neq y \\ x,y \in C}} \delta(x, y)\,.$$

Remark 2.6 *The hamming distance δ is a metric on strings of a fixed length, and in particular satisfies the triangle inequality.*

The distance can be used to understand the number of errors one can correct, as follows.

Proposition 2.7 *A code $C : \mathbb{F}_p^k \to \mathbb{F}_p^n$ can correct t errors if and only if $\Delta(C) \geq 2t + 1$.*

Note that for the code in Example 2.4, $\Delta(C) = 3$. This implies that the code from Example 2.4 can handle no more than one error. We now prove the bound:

Proof: First, we prove the reverse direction. If $\Delta(C) \geq 2t+1$, then $\frac{\Delta(C)}{2} > t$. For each codeword $x \in C$ let $H(x,r) = \{y \in \mathbb{F}_p^n \ : \ \delta(x,y) < r\}$. In particular consider the case when we let $r = \frac{\Delta(C)}{2}$. Notice that for two distinct $x, x' \in C$, $H(x,r) \cap H(x',r) = \emptyset$, since otherwise, $\delta(x,x') < \Delta(C)$. Now, suppose a codeword is corrupted in t positions to the string y. Since $t < \frac{\Delta(C)}{2}$, we have that the corrupted message y is contained in precisely one set $H(x,r)$. Thus, we can simply find the unique x such that $y \in H(x,r)$.

Now, we prove the forward direction. Suppose the codeword $x \in C$ is corrupted in t bits to y i.e., $\delta(x,y) = t$. Since we can correct up to t errors, we know that we can find $w \in C$ such that $w = argmin_{z \in C}(\delta(y,z))$. Moreover, we want w to be equal to z. Thus, we have that

$$\delta(z,y) \ \geq \ \delta(x,y) \ = \ t \quad \forall z \in C\,.$$

However, if we have x, z as the closest codewords in C, we can always find y such that $\delta(x,y) = t$ and $\delta(z,y) = \Delta(C) - t$. Combining it with the above, we get $\Delta(C) - t > t$, which implies $\Delta(C) \geq 2t+1$. ■

References

[Wig17] Avi Wigderson. Operator scaling: theory, applications and connections, 2017. Tutorial given at CCC 2017. URL: http://computationalcomplexity.org/Archive/2017/tutorial.php. 2