CAS CS 599 B: Mathematical Methods for TCS

Lecturer: Mark Bun Spring 2022

Lecture Notes 19:

Error-Correcting Codes, Linear Codes

Reading.

- Guruswami-Rudra-Sudan §1, 2

Today we'll start talking about coding theory. A motivating scenario for this area is the following communication problem. Suppose a sender wishes to transmit a message x to a receiver. However, communication can only take place across a noisy channel. To deal with noise, the sender encodes x into a codeword c before sending it across the channel. The receiver obtains a corrupted codeword $\hat{c}$ and would like to confidently decode it back to the original message x. How can we design codes that minimize redundancy, tolerate large numbers of errors, and have efficient encoding and decoding algorithms?

Coding theory is studied from a number of different perspectives. Modeling the channel as introducing random noise from a known distribution (e.g., the binary symmetric channel) is traditional in information theory and electrical engineering. Meanwhile, computer scientists typically focus on the worst-case setting where errors can be introduced arbitrarily by an adversary. Both models are interesting and important, with plenty of applications in theoretical CS. Since I have to make hard choices and stop somewhere, I'll primarily talk about worst-case errors.

1 Error-Correcting Codes

- Let Σ be a finite alphabet, with $q = |\Sigma|$. If I don't say otherwise, $\Sigma = \{0, 1\}$ and $q = 2$.
- A *code* of block length n over alphabet Σ is a subset $C \subseteq \Sigma^n$.

Associated to a code is two functions, both of which we would like to be computable in polynomial time by the sender and receiver, respectively.

- $\mathrm{Enc} : M \to C$ is an injective function that takes a message to a codeword.
- $\mathrm{Dec} : \Sigma^n \to M$ takes a corrupted codeword to a message.

Let's consider the simple problem of correcting for a single adversarially chosen error on the message space $M = \{0, 1\}^4$.

Example 1. The repetition code $C = \{(x_1, x_1, x_1, \ldots, x_4, x_4, x_4) \mid x \in \{0, 1\}^4\}$. This comes with the natural encoding function $\mathrm{Enc}(x) = (x_1, x_1, x_1, \ldots, x_4, x_4, x_4)$. To decode from one error, take the majority vote of every block of 3 bits.

Example 2. The Hamming code maps 4 bits to 7 bits as follows: $\mathrm{Enc}(x_1, x_2, x_3, x_4) = (x_1, x_2, x_3, x_4, x_1 \oplus x_2 \oplus x_4, x_1 \oplus x_3 \oplus x_4, x_2 \oplus x_3 \oplus x_4)$. You can check that this corrects for one error by inspecting cases. We'll see a more general framework for analyzing codes like this in a bit.

The Hamming code encodes 4-bit messages using a much shorter codeword than the repetition code. We measure the redundancy of a code by its rate:

Definition 3. Let $C \subseteq \Sigma^n$ be a code.

- The *dimension* of C is $k = \log_q |C|$. A code of dimension k corresponds to message space Σ^k.
- The *rate* of C is $R = \frac{k}{n}$.

In general, we'd like to design codes that can correct for more than 1 error. Define a *t-error-correcting code* to be a code C such that for every $x \in M$ and every v such that $\Delta(v, \mathrm{Enc}(x)) \leq t$, we have $\mathrm{Dec}(v) = x$. Here, Δ denotes the Hamming distance: the number of positions on which two strings differ.

The error tolerance of a code has a nice geometric characterization. [Draw a picture.] Namely, a code C is t-error correcting if and only if the Hamming balls around each codeword of radius t are all disjoint. This, in turn, happens if and only if the codewords themselves are far apart in Hamming distance. This motivates the following definition.

Definition 4. Let $C \subseteq \Sigma^n$ be a code. The (minimum) distance of C, denoted $d(C)$ is defined by

$$d(C) = \min_{v,w \in C} \Delta(v, w).$$

If C has distance d, then it is $\lfloor (d-1)/2 \rfloor$-error correcting. High distance codes are error tolerant according to other natural measures as well. For example, a distance d code enables *detection* of up to $d-1$ errors, as well as the correction of up to $d-1$ erasures.

A basic problem in coding theory is to understand what tradeoffs between rate and distance are achievable. The Hamming code has rate $4/7$ and distance 3. It turns out that the Hamming code is a *perfect* code: the balls of radius 1 around each codeword exactly partition $\{0,1\}^7$, and hence it has optimal rate among all distance 3 codes.

2 Linear Codes

As is usual for combinatorial constructions, one can show that near-optimal codes exist by taking a random subset of Σ^n or constructing a packing of Σ^n greedily. Codes like this take exponential space to describe. In applications, we'd like explicit codes which have polynomial-time computable encoding. The most important class of explicit codes are linear codes, for which efficient encoding is immediate.

Definition 5. Let q be a prime power and let $\Sigma = \mathbb{F}_q$ be a finite field. A *linear code* is a subspace $C \subseteq \mathbb{F}_q^n$. If C has dimension k and distance d, then we refer to it as a $[n, k, d]_q$ code. Sometimes we write $[n, k]_q$ when we want to suppress the distance.

A k-dimensional subspace, and hence a linear code, can be described as the row span of a $k \times n$ matrix. That is, there exists a rank-k matrix $G \in \mathbb{F}_q^{k \times n}$ called the *generator matrix* for C such that

$$C = \{xG \mid x \in \mathbb{F}_q^k\}.$$

Note the somewhat unusual convention here that we are interpreting messages and codewords as **row vectors**. To encode a message x, one simply takes $\mathrm{Enc}(x) = xG$.

Example 6. The Hamming code is a linear code over $\mathbb{F}_2^7$. It has generator matrix

$$G_{\text{Ham}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Now let's consider the error detection problem for linear codes. Given a vector $v \in \mathbb{F}_q^n$, how can we recognize whether v is exactly a codeword? We can do this using the "dual" representation of a k-dimensional subspace as the kernel of a rank-$(n-k)$ matrix. Specifically, if C has dimension k, then there exists a rank-$(n-k)$ matrix $H \in F_q^{(n-k)\times n}$ called the *parity check matrix* for C such that

$$C = \{v \in \mathbb{F}_q^n \mid Hv^T = \mathbf{0}\}.$$

Example 7. A parity check matrix for the Hamming code is

$$H_{\text{Ham}} = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

The columns of the parity check matrix for the Hamming code are simply all 7 nonzero binary column vectors over $\mathbb{F}_2^3$. Permuting the columns to be in a systematic order, one can generalize this construction to a family of $[2^r-1, 2^r-r-1, 3]_2$-Hamming codes with parity check matrix

$$\begin{pmatrix} 0 & 0 & \dots & 1 & \dots & 0 & 1 \\ & & & \vdots & & & \\ 0 & 1 & \dots & 0 & \dots & 1 & 1 \\ 1 & 0 & \dots & 0 & \dots & 1 & 1 \end{pmatrix}.$$

The distance of a linear code is very clean to characterize:

Fact 8. *The minimum distance $d(C)$ of a code is the minimum Hamming weight of a nonzero codeword.*

Proof. Since a linear code is a subspace, if v, w are codewords, then $v - w$ is also a codeword. The fact follows from observing that $\Delta(v, w) = \Delta(v - w, \mathbf{0})$. □

Fact 9. *The minimum distance $d(C)$ of a code with parity check matrix H is the minimum number of columns of H that are linearly dependent.*

Proof. By Fact 8, $d(C)$ is the minimum weight of a vector v such that $Hv^T = \mathbf{0}$. This is exactly the minimum number of columns of H such that some linear combination of those columns is $\mathbf{0}$. □

Example 10. To see that the Hamming code has distance 3, you can check that every pair of columns is distinct, hence linearly independent. However, columns 3, 4, and 7 are linearly dependent.

2.1 Decoding the Hamming code

The parity check matrix for the Hamming code gives a very efficient method for correcting a single error. Suppose $v = \text{Enc}(x)$ is a codeword that is corrupted by an error vector e. Then

$$H(v+e)^T = Hv^T + He^T = He^T.$$

If $e = \mathbf{0}$, there is no error and this is the zero vector. So we can recover the message x by solving the linear system $xG = v$. On the other hand, if $e = e_i$ where i is the index of the flipped bit, then He_i^T is the i'th column of H_{Ham}. We can then just subtract this off from v before solving for x.

3 Dual Codes

The dual of a code C, denoted by $C^{\perp}$ is the subspace orthogonal to C. That is,

$$C^{\perp} = \{w \mid \langle w, v \rangle = 0 \text{ for all } v \in C\}.$$

If C is an $[n, k]_q$ code, then $C^{\perp}$ is an $[n, n-k]_q$ code.

By definition, if G is a generator matrix for C, then G is a parity check matrix for $C^{\perp}$. Similarly, if H is a parity check matrix for C, then H is a generator matrix for $C^{\perp}$.

The dual code of the $[2^r - 1, 2^r - r - 1, 3]_2$-Hamming code has H_{Ham} as its generator matrix. If we prepend the all-zeroes column to this matrix, we get

$$G_{\text{Had}} = \begin{pmatrix} 0 & 0 & 0 & \dots & 1 & \dots & 1 & 1 \\ 0 & & & & \vdots & & & \\ 0 & 0 & 1 & \dots & 0 & \dots & 1 & 1 \\ 0 & 1 & 0 & \dots & 0 & \dots & 0 & 1 \end{pmatrix}.$$

This is the generator matrix for the $[2^r, r]_2$-Hadamard code. A useful way to think about this code is via its encoding function $\mathrm{Enc}(x) = (\langle x, a \rangle)_{a \in \mathbb{F}_2^r}$. To encode a message x, just take the inner product with all vectors in $\mathbb{F}_2^r$. This has terrible rate (k is only logarithmic in n), but excellent distance.

Fact 11. *The $[2^r, r]_2$-Hadamard code has distance 2^{r-1}.*

Proof. By Fact 8, it's enough to show that the minimum Hamming weight of a nonzero codeword is 2^{r-1}. This is true for the same reasons that distinct Fourier characters on the hypercube are orthogonal, i.e., $(\chi_S(a))_{a \in \{-1,1\}^r}$ agrees with $(\chi_T(a))_{a \in \{-1,1\}^r}$ in exactly 2^{r-1} locations for every $S \neq T$. $\square$