11 Section Week 11 - Boolean Functions and Fourier Analysis

Based roughly on Ryan O'Donnell's text "Analysis of Boolean Functions."

This lecture assumes knowledge of matrix multiplication, and in the notes doesn't include a description of the Kronecker product.

We'll continue where we left off last time with fourier analysis. To review:

11.1 Fourier Analysis of Boolean Functions

The idea of Fourier analysis is to interpolate a boolean function with a polynomial over the real numbers. We view booleans as $\{\pm 1\}$ instead of $\{0, 1\}$ and a function as $f : \{\pm 1\}^n \to \mathbb{R}$. This allows us to interpolate (say, a function over 3 variables) as

$$\begin{aligned}
&\left(\frac{1}{2} - \frac{1}{2}x_1\right)\left(\frac{1}{2} - \frac{1}{2}x_2\right)\left(\frac{1}{2} - \frac{1}{2}x_3\right) \cdot f(-1, -1, -1) + \\
&\qquad\qquad\qquad\qquad\qquad\qquad\vdots \\
&\left(\frac{1}{2} + \frac{1}{2}x_1\right)\left(\frac{1}{2} + \frac{1}{2}x_2\right)\left(\frac{1}{2} + \frac{1}{2}x_3\right) \cdot f(+1, +1, +1)
\end{aligned}$$

Once we take a polynomial like this and FOIL it out, we get a multivariate polynomial, which is also multilinear because the exponent for each variable is at most 1, of the form:

Theorem (Multilinear Representation of Boolean Functions)**.** *Every boolean function $f : \{\pm 1\}^n \to \mathbb{R}$ is uniquely expressible as a multilinear polynomial of the form*

$$\sum_{S \subseteq [n]} \hat{f}(S) \prod_{i \in S} x_i$$

where $\hat{f}(S)$ is a coefficient determined by the function f. This is called the "Fourier Expansion" of f, and $\hat{f}(S)$ is called the Fourier coefficient of S. Notice that the product is just the parity of set S, so this is an expression of f as a linear combination of parity functions over subsets of $[n]$.

Why is this equivalent to our function? In particular, why are we summing over all subsets S of the indices $[n]$?

Work out examples like Maj and parity

We can rewrite $\prod_{i \in S} x_i$ as $\chi_S(x)$, the parity function for the indices in S. So where fourier transform of signals is decomposing a signal into a combination of sines and cosines, what we're doing here is expressing a function as a linear combination of parity functions.

The reason why we care about this representation is that it turns out that the fourier coefficients tell us interesting things about the behavior of the function.

11.2 Computing a Function from its Fourier Coefficients

Suppose we have a lot of coefficients for a multilinear polynomial and wish to know which boolean function the polynomial is representing. We can get the truth table for the boolean function by matrix multiplication:

$$\begin{bmatrix} f(+1 \cdots +1) \\ \vdots \\ f(-1 \cdots -1) \end{bmatrix} = \begin{bmatrix} \chi_\emptyset(+1 \cdots +1) & \chi_{\{1\}}(+1 \cdots +1) \ldots \chi_{[n]}(+1 \cdots +1) \\ \vdots & \\ \chi_\emptyset(-1 \cdots -1) & \chi_{\{1\}}(-1 \cdots -1) \ldots \chi_{[n]}(-1 \cdots -1) \end{bmatrix} \times \begin{bmatrix} \hat{f}(\emptyset) \\ \vdots \\ \hat{f}([n]) \end{bmatrix}$$

The big χ matrix is called the **Hadamard Matrix** H_N, where $N = 2^n$ and $H_N[x, S] = \chi_S(x)$. It is of the form $H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ and in general $H_N = H_2 \otimes H_2 \cdots \otimes H_2$ n times ($\otimes$ is the Kronecker product).

This is an important matrix and you'll definitely see it again. Kronecker product is also important, and we can stop to discuss it.

Expressing H_N in this way allows us to compute

$$H_N \begin{bmatrix} \hat{f}(\emptyset) \\ \vdots \\ \hat{f}([n]) \end{bmatrix}$$

in $O(N \log N)$ time by divide and conquer. This is called the **Fast Walsh-Hadamard Transform**.

Show for Maj function

We make two observations about the Hadamard matrix:

1. The dot product of two different columns of H_N is 0 (remember this for later)
2. The dot product of a column of H_N with itself is N

These facts give that $\frac{1}{\sqrt{N}} H_N$ is unitary: $\frac{1}{N}\left(H_N H_N^\top\right) = I$. Since H_N is symmetric, $H_N = H_N^\top$ and $\frac{1}{N}\left(H_N H_N\right) = I$.

Thus, since

$$\begin{bmatrix} f(+1 \cdots +1) \\ \vdots \\ f(-1 \cdots -1) \end{bmatrix} = H_N \begin{bmatrix} \hat{f}(\emptyset) \\ \vdots \\ \hat{f}([n]) \end{bmatrix}$$

If we have the truth table and want to get the fourier coefficients, we can just multiply on the left of our truth table by $\frac{1}{N} H_N$.

$$\frac{1}{N} H_N \begin{bmatrix} f(+1 \cdots +1) \\ \vdots \\ f(-1 \cdots -1) \end{bmatrix} = \begin{bmatrix} \hat{f}(\emptyset) \\ \vdots \\ \hat{f}([n]) \end{bmatrix}$$

So we can write

$$\begin{aligned} \hat{f}(S) &= \frac{1}{N} \sum_{x \in \{\pm 1\}^n} f(x)\chi_S(x) \\ &= 2^{-n} \sum_{x \in \{\pm 1\}^n} f(x)\chi_S(x) \\ &= \mathbb{E}_{x \leftarrow_R \{\pm 1\}^n}[f(x)\chi_S(x)] \end{aligned}$$

11.3 Application to Probability

For functions f, g, write

$$\langle f, g \rangle = 2^{-n} \sum_x f(x)g(x)$$

or using expected value notation $\mathbb{E}_{x \leftarrow_R \{\pm 1\}^n}[f(x)g(x)]$. So in some sense, this is the "correlation" between functions f and g: if they agree a lot, this number is high (agreements multiply to 1, disagreements multiply to -1).

If we replace f, g with their Fourier expansions we take advantage of linearity of vectors and recall that $\langle \chi_S, \chi_T \rangle$ is 0 for $S \neq T$ we get:

$$\begin{aligned}\langle f, g\rangle &= \left\langle \sum_S \hat{f}(S)\chi_S, \sum_T \hat{g}(T)\chi_T \right\rangle \\ &= \sum_{S,T} \hat{f}(S)\hat{g}(T)\left\langle \chi_S, \chi_T\right\rangle \\ &= \sum_S \hat{f}(S)\hat{g}(S)\end{aligned}$$

Thus, comparing to the original definition somehow the similarity of the functions f, g can be determined both by stacking their truth tables together or stacking their Fourier coefficients together. The similarity of the functions is identical to the similarity of their fourier coefficients.

Since $\hat{f}(S) = \langle f, \chi_S\rangle$, observe that $\hat{f}(\emptyset) = \langle f, 1\rangle = \mathbb{E}[f(x)]$ and is thus the average value of the boolean function.

This is the most simple and direct "application" of Fourier coefficients

Also as a corollary we get $\langle f, f\rangle = \mathbb{E}_x[f(x)^2] = 1$ if f is boolean-valued (i.e. image is $\{\pm 1\}$, so

$$\sum_{S\subseteq[n]} \hat{f}(S)^2 = 1$$

Furthermore, the variance of f is

$$\mathbf{Var}_x[f(x)] = \mathbb{E}[f(x)^2] - \mathbb{E}[f(x)]^2 = 1 - \hat{f}(\emptyset)^2 = \sum_{S\neq\emptyset} \hat{f}(S)^2$$

11.4 Application to Social Choice

Consider a boolean valued function f to be a "voter rule" in an n voter, 2 candidate election. Democracy is majority function, America is some electoral college function, dictatorship is $f(x) = x_i$.

Influence: influence of ith voter is

$$\Pr_{x \leftarrow_R \{\pm 1\}^n} [f(x_0, \ldots x_i, \ldots x_n) \neq f(x_0, \cdots - x_i, \ldots x_n)]$$

It turns out that influence of i is

$$\sum_{S \subseteq [n], i \in S} \hat{f}(S)^2$$

Using analysis of boolean functions we can also prove

- Arrow's Theorem: The only voting rule that works for 3 or more candidates is dictatorship, and the proof involves Candorcet's paradox
- KKL (Kahn, Kalai, and Linial) theorem: given any 2 candidate voting rule with $\mathbb{E}[f(x)] = 0$ (so f is not inherently biased) we can always find $o(1)$ fraction of the voters where if we fix those votes we can fix the outcome of the election with probability $1 - o(1)$