The Harmonic Analysis of Boolean Functions

Jacob Denson

March 17, 2017

Table Of Contents

Chapter 1

Introduction to Boolean Analysis

1.1 Introduction

This course is about the theory of discrete harmonic analysis. The rough idea is that if we are trying to understand some set X with a 'binary representation', in the sense that it can be put into one to one correspondence with $\{0,1\}^n$, then the abelian group structure on $\{0,1\}^n$ induces a natural group structure on X, and we can obtain powerful results about X through the Fourier transform corresponding to the group operation, provided the induced group operation is meaningful to the structure of X. In particular, Fourier analysis is particularly powerful at obtaining quantitative results about real-valued functions $f : X \to \mathbf{R}$ and it's translations $(\sigma_y f)(x) = f(x+y)$, because the characters on X simultaneously diagonalize the translation operators σ_y.

Example. *The Hamming distance between two length n binary strings x and y is the number of indices where the strings differ, which we denote by*

$$\Delta(x,y) = \#\{i : x^i \neq y^i\}$$

Then $\Delta(x,y)$ is the L^1 norm, viewing elements of $\{0,1\}^n$ as $\{0,1\}$ indicator functions on $\{1,\ldots,n\}$. If we want to quantify $f(x) - f(y)$ in terms of $\Delta(x,y)$, for some $f : \{0,1\}^n \to \mathbf{R}$, then Fourier analysis becomes applicable, because $f(x) - f(y) = (\sigma_x - \sigma_y)f(0)$. Essentially all the problems and techniques considered in this book can be reduced to an analysis of the Hamming distance between bit strings.

Example. *Given a set of n vertices $x_1, \ldots, x_n$, the set of all directed graphs on the vertices $x_1, \ldots, x_n$ can be identified with elements of $\{0,1\}^{n \times n}$, where the (i,j)'th bit of a string corresponds to whether the edge (x_i, x_j) is in the particular graph. The Hamming distance between two graphs is then the number of edges upon which the graphs differ. We may want to consider how quantities on graphs change under this Hamming distance, such as the shortest path between points, or the minimum cut, and Fourier analysis gives a basic set of techniques to quantify this change.*

Example. *A subset of a set with n elements can be identified with a Boolean string in $\{0,1\}^n$, which we can view as an indicator function which identifies which elements are present in the subset. The Hamming distance then measures the cardinality of the symmetric difference between two subsets, and this explains why Fourier analytic methods have applications in combinatorial problems in set theory.*

Example. *The truth values $\top$ and $\bot$ have an obvious relation to $\{0,1\}$, especially if you've used a modern programming language. The group operation on $\{0,1\}$ then induces an operation on $\top$ and $\bot$, which is the exclusive or operation*

$$A \oplus B = \begin{cases} \top & A \text{ is } \top, \text{ or } B \text{ is } \top, \text{ but not both.} \\ \bot & \text{otherwise} \end{cases}$$

and where multiplication is conjunction. Given a property P of elements of some set X, this relation is used to form the indicator $\mathbf{I}(P(x)) : X \to \{0,1\}$, and then Fourier analytic methods can be used to analyze 'true Boolean functions' $f : \{\top, \bot\}^n \to \{\top, \bot\}$.

The most important correspondence with the additive structure of $\{0,1\}$ in the general theory will be the multiplicative group $\{-1,1\}$, where we map 0 to 1, and 1 to -1. Initially, this is a very strange correspondence to pick, because we normally think of 1 as being the value of 'truth'. However, the correspondence does give an isomorphism between the two group structures, and fits with the general convention of performing harmonic analysis over the multiplicative group $\mathbf{T}^n$ where possible. Rather than thinking of $\{0,1\}$ as the canonical group upon which we take the correspondence, we shall find that $\{-1,1\}$ will become a much more central figure in the Fourier analysis.

For instance, the characters on $\{-1,1\}$ are notationally much more natural than the characters on $\mathbf{F}_2^n$. The characters on the group $\mathbf{T}^n$ take the

form of integer-valued monomials $z_1^{m_1} \dots z_n^{m_n}$. Because $\{-1,1\}^n$ embeds as a closed subgroup of $\mathbf{T}^n$, every character on $\{-1,1\}^n$ is the restriction of a character on $\mathbf{T}^n$, and can therefore be represented by a monomial. Since $x_i^2 = 1$ for all $x_i \in \{-1,1\}$, these monomials can be chosen with $m_i \in \{0,1\}$. Abstract harmonic analysis tells us that there are exactly 2^n characters on the group $\{-1,1\}$, so these are all the identifications we can make, and therefore we conclude that the functions

$$x^S = \prod_{i \in S} x^i$$

form the set of all characters on $\{-1,1\}$, for $S \subset \{1,\dots,n\}$. Using functional notation and switching domains, the $\mathbf{F}_2$-valued characters on $\mathbf{F}_2^n$ are

$$\sigma_S(x) = \sum_{i \in S} x_i$$

and the complex-valued characters from $\mathbf{F}_2^n$ are

$$\chi_S(x) = (-1)^{\sigma_S(x)} = (-1)^{\sum_{i \in S} x_i}$$

The basic theory of discrete abelian harmonic analysis tells us that the characters x^S form an orthonormal basis for the Hilbert space of real-valued functions on $\mathbf{F}_2^n$, so for any $f : \{-1,1\}^n \to \mathbf{R}$, $g : \mathbf{F}_2^n \to \mathbf{R}$, we have unique expansions

$$f(x) = \sum_S \widehat{f}(S) x^S \qquad g(x) = \sum_S \widehat{g}(S) \chi_S(x)$$

where $\widehat{f}(S), \widehat{g}(S) \in \mathbf{R}$ are the *Walsh-Fourier coefficients* corresponding to S. We shall also find it is natural to consider the Fourier transforms of functions on vector spaces $\mathbf{F}_2^I$ and $\{-1,1\}^I$, where I is an arbitrary index set, in which the characters can be represented as x^S and χ_S, for $S \subset I$, and the expansion takes the form

$$f(x) = \sum_{S \subset I} \widehat{f}(S) x^S$$

We will occassionally use this notation when it is more natural to do so.

An easy way to find Fourier coeffients, without any harmonic analysis, is to expand the any function $f : \{-1,1\}^n \to \{-1,1\}$ in its conjunctive normal form[1], calculating

$$f(x) = \sum_y \mathbf{I}(x = y) f(y) = \sum_y \frac{1}{2^n} \left(\prod_{i=1}^n 1 + x_i y_i \right) f(y)$$

This formula gives us a *particular* expansion of the function, and we know it must be the *unique* expansion. This is essentially a binary equivalent to the technique used to find power series of holomorphic functions, combining well know expansions to form the expansion of a particular formula.

Example. *The maximum function* $\max(x,y)$ *on* $\{-1,1\}^2$ *has an expansion*

$$\max(x,y) = \frac{1}{2}(1 + x + y - xy)$$

which corresponds to the the conjunction function $f(x,y) = x \wedge y$ *on* $\{\top, \bot\}^2$*, or the minimum function on* $\mathbf{F}_2^2$*. To obtain this expansion, and a more general expansion for the maximum function* $\max_n$ *on n variables note that we can write*

$$\mathbf{I}(x = -1) = \frac{1}{2^n} \prod_{i=1}^n (1 - x_i) = \frac{1}{2^n} \sum_S (-1)^{|S|} x^S$$

so that

$$\begin{aligned} \max(x_1, \ldots, x_n) &= \mathbf{I}(x \neq -1) - \mathbf{I}(x = -1) \\ &= 1 - 2\mathbf{I}(x = -1) \\ &= \left(1 - \frac{1}{2^{n-1}}\right) - \frac{1}{2^{n-1}} \sum_{S \neq \varnothing} (-1)^{|S|} x^S \end{aligned}$$

[1] If a property on $\{-1,1\}^n$ is identified with it's indicator function, then the indicator function of $P \wedge Q$, for any two properties P and Q, is obtained by multiplying together the corresponding indicator functions. Since the property $x_i = y_i$ has indicator function $\mathbf{I}(x_i = y_i) = (1/2)(1 + x_i y_i)$, then we see that the expansion obtained for general functions $f : \{-1,1\}^n \to \{0,1\}$ is exactly the one induced by the canonical conjunctive form expression $P(x) = \bigvee_{P(y)} (y = x) = \bigvee_{P(y)} (\bigwedge_{i=1}^n y_i = x_i)$, and we have extended this representation to general functions $f : \{-1,1\}^n \to \mathbf{R}$.

Example. *Consider the minimum function* $\min(x) : \{-1,1\}^n \to \{-1,1\}$ *on n bits, which corresponds to the disjunction operation on* $\{\top,\bot\}^n$*. Since we have the relationship* $\min(x) = -\max(-x)$*, we find*

$$\begin{aligned} -\max(-x) &= -\left[\left(1 - \frac{1}{2^{n-1}}\right) - \frac{1}{2^{n-1}} \sum_{S \neq \emptyset} (-1)^{|S|}(-x)^S\right] \\ &= \left(\frac{1}{2^{n-1}} - 1\right) + \frac{1}{2^{n-1}} \sum_{S \neq \emptyset} x^S \end{aligned}$$

and this gives the expansion for $\min(x)$*.*

Note that the minimum and maximum functions on n variables are $1/2^{n-1}$ close to constant functions in the L^1 distance, hence the Fourier transformations of the minimum and maximum functions are $1/2^{n-1}$ close to the Fourier transforms of constant functions 1 and -1 in the L^∞ distance. Thus we see that the Fourier coefficients of min and max are bounded in absolute value by $1/2^{n-1}$, and the expected values of the functions are within a distance $1/2^{n-1}$ from 1 and -1.

Example. *In general, the indicator functions* $\mathbf{I}(x = a) : \{-1,1\}^n \to \{0,1\}$*, for some* $a \in \{-1,1\}^n$ *can be expressed as*

$$\mathbf{I}(x = a) = \frac{1}{2^n} \prod_{i=1}^{n} (1 + x_i a_i) = \frac{1}{2^n} \sum_S a^S x^S$$

If we only have a subset $X = \{i_1, \ldots, i_k\}$ *of indices we want to specify, then*

$$\mathbf{I}(x_{i_1} = a_1, \ldots, x_{i_k} = a_k) = \frac{1}{2^k} \sum_{S \subset X} a^S x^S$$

Thus the degree of the indicator function is bounded by k: the only nonzero Fourier coefficients correspond to sets S with cardinality bounded by k.

Example. *Consider n Bernoulli distributions over* $\{-1,1\}$ *with means* $\mu_1, \ldots, \mu_n$*, and consider the corresponding product distribution on* $\{-1,1\}^n$*, with density function* f*. If X is a random variable with this distribution, then*

$$\mathbf{P}(X_i = 1) - \mathbf{P}(X_i = -1) = 2\mathbf{P}(X_i = 1) - 1 = \mu_i$$

so $\mathbf{P}(X_i = 1) = \frac{1}{2}(\mu_i + 1) = P_i \in [0,1]$. *We then calculate*

$$\begin{aligned} f(x_1,\ldots,x_n) &= \prod_{i=1}^{n} [P_i \mathbf{I}(x_i = 1) + (1 - P_i)\mathbf{I}(x_i = -1)] \\ &= \sum_a \left(\prod_{a_i=1} P_i \right) \left(\prod_{a_i=-1} (1 - P_i) \right) \mathbf{I}(x = a) \\ &= \frac{1}{2^n} \sum_S \sum_a a^S \left(\prod_{a_i=1} P_i \right) \left(\prod_{a_i=-1} (1 - P_i) \right) x^S \\ &= \frac{1}{2^n} \sum_S \left(\prod_{i \in S} (2P_i - 1) \right) x^S \\ &= \frac{1}{2^n} \sum_S \mu^S x^S \end{aligned}$$

where we obtain the second last equation by repeatedly factoring out the coordinates a_i *for* $a_i = 1$ *and* $a_i = -1$.

Example. *Define the 'inner product' function* f *on* $\mathbf{F}_2^n \times \mathbf{F}_2^n$ *by*

$$f(x,y) = (-1)^{\langle x,y \rangle} = \prod_{i=1}^{n} (-1)^{x_i y_i}$$

which is a bilinear map from $\mathbf{F}_2$ *to* $\{-1,1\}$. *This corresponds to the function* g *on* $\{-1,1\}^n \times \{-1,1\}^n$ *defined by*

$$g(x,y) = \prod_{i=1}^{n} (-1)^{\mathbf{I}(x_i = y_i = -1)}$$

For $n = 1$, *we have*

$$g(x,y) = \begin{cases} -1 & x = y = -1 \\ 1 & \textit{otherwise} \end{cases}$$

Hence $g(x,y)$ *is just the* max *function on* $\{-1,1\}^2$, *and we find*

$$g(x,y) = (-1)^{\mathbf{I}(x=y=-1)} = \frac{1}{2}(1 + x + y - xy)$$

In general, any subset of indices on $\{1,2\}\times[n]$ (this is the index set for $\{-1,1\}^n\times\{-1,1\}^n$) can be decomposed as $S = \{1\}\times S_1 \cup \{2\}\times S_2$, where $S_1, S_2 \subset [n]$. Let $\alpha(S)$ be the cardinality of $S_1 \cap S_2$. Then we have

$$\begin{aligned} g(x,y) &= \frac{1}{2^n}\prod_{i=1}^{n}(1+x_i+y_i-x_iy_i) \\ &= \frac{1}{2^n}\sum_{S\subset[n]}(-1)^{|S|}(xy)^S(1+x+y)^{S^c} \\ &= \frac{1}{2^n}\sum_{S\subset[n]}(-1)^{|S|}\sum_{T\subset S^c}(xy)^S(x+y)^T \\ &= \frac{1}{2^n}\sum_{S\subset\{1,2\}\times[n]}(-1)^{\alpha(S)}(x,y)^S \end{aligned}$$

Note that

$$g(x,x) = \frac{1}{2^n}\sum_{S}(-1)^{\alpha(S)}x^{S_1\triangle S_2}$$

The number of S_1 and S_2 with $S_1\triangle S_2 = [n]$ is exactly the number of bipartitions of $\{1,\dots,n\}$, of which we have 2^n elements, and these partitions must be disjoint, hence the corresponding $\alpha(S)$ is always equal to 0, hence $x^{[n]}$ has a coefficient of 1. Conversely, if T is a proper subset of $[n]$, not containing some element x, then the set of S_1, S_2 can be divided in half into the family such that $x \in S_1$ and $x \in S_2$ and the family such that $x \notin S_1 \cup S_2$. The coefficient $(-1)^{\alpha(S)}$ on one family is the opposite of the coefficient of the other family, so when we sum over all possible coefficients, we find that the coefficient of $g(x,x)$ corresponding to x^T is zero. This makes sense, since $g(x,x) = x^{[n]}$.

Example. *The equality function EQ returns 1 only if all x_i are equal. Thus*

$$\begin{aligned} EQ(x) &= \mathbf{I}(x_1 = x_2 = \cdots = x_n = 1) + \mathbf{I}(x_1 = x_2 = \cdots = x_n = -1) \\ &= \frac{1}{2^n}\sum_{S}(1+(-1)^{|S|})x^S = \frac{1}{2^{n-1}}\sum_{|S|\ even}x^S \end{aligned}$$

which makes sense since EQ is an even function, so the odd degree coefficients vanish. The not all equal function NAE returns 1 only if all x_i not all equal. Then

$$NAE(x) = 1 - EQ(x) = \left(1-\frac{1}{2^{n-1}}\right) - \sum_{\substack{|S|\ even \\ S\neq\varnothing}} x^S$$

We will find this function is very useful in analyzing the effectiveness of voting rules in chapter 2.

Example. *The sortedness function* $Sort : \{-1,1\}^4 \to \{-1,1\}$ *returns -1 if the bits in the input are monotonically increasing or monotonically decreasing. Since the function is even, all odd Fourier coefficients vanish. It will also help that Sort is invariant under the coordinate permutation* $(14)(23)$*, hence the Fourier coefficients obtained by swapping these coordinates are equal. To calculate the coefficients, it will be helpful to switch to calculating the coefficients of* $f = (1 - Sort)/2$*, which is now* $\{0,1\}$ *valued.* $f(x) = 1$ *if and only if x is one of the following 8 bit strings*

$$(-1,-1,-1,-1),(-1,-1,-1,+1),(-1,-1,+1,+1),(-1,+1,+1,+1)$$

$$(+1,+1,+1,+1),(+1,+1,+1,-1),(+1,+1,-1,-1),(+1,-1,-1,-1)$$

Hence the expected value of f *is* $1/2$*. We also calculate that*

$$\widehat{f}(\{1,2,3,4\}) = 0 \quad \widehat{f}(\{1,2\}) = \widehat{f}(\{3,4\}) = 1/4$$

$$\widehat{f}(\{1,3\}) = \widehat{f}(\{2,4\}) = 0 \quad \widehat{f}(\{1,4\}) = -1/4$$

$$\widehat{f}(\{2,3\}) = 1/4$$

Thus $f(x) = (1/4)(2 + x_1x_2 + x_3x_4 + x_2x_3 - x_1x_4)$*, and so*

$$Sort(x) = 1 - 2f(x) = (1/2)(x_1x_4 - x_1x_2 - x_2x_3 - x_3x_4)$$

The x_1x_4 *term determines whether x should be the constant* 1 *or* -1 *term, and the rest offset the problem by pairwise comparisons.*

Example. *The hemi-icosohedron function* $HI : \{-1,1\}^6 \to \{-1,1\}$ *takes a set of labels for the vertices of the hemi-icosohedron graph below, and returns the number of faces labelled* $(1,1,1)$*, minus the number of faces labelled* $(-1,-1,-1)$*, modulo 3*

(INSERT PICTURE OF HEMI-ICOSOHEDRON HERE)

The graph is homogenous, in the sense that there is a graph isomorphism mapping a vertex to any other vertex. In fact, one can obtain all isomorphisms by choosing which face one face will be mapped to, and specifying where two of the vertices on that face will go in the new face.

For any x, if we consider $-x$, then any face labelled $(1,1,1)$ becomes a face labelled $(-1,-1,-1)$, and vice versa, so that $HI(-x) = -HI(x)$. Thus HI is an odd function, and all even Fourier coefficients vanish. Note that $HI(-1) = -1$, because all 10 faces are labelled $(-1,-1,-1)$, and $H(1) = 1$. If x has a single 1, then 5 of the faces are labelled $(-1,-1,-1)$, and none are labelled $(1,1,1)$, so $HI(x) = 1$. If x has exactly 2 ones, then we see that only two faces are labelled $(-1,-1,-1)$, and none are labelled $(1,1,1)$, so $HI(x) = 1$. If x has three ones arranged in a triangle on the hemi-icosohedron, then the hemi-icosohedron has one face labelled $(1,1,1)$, and none labelled $(-1,-1,-1)$, and hence $HI(x) = 1$. If x does not have three ones arranged in a triangle, then we find it must have three negative ones arranged in a triangle, hence $HI(x) = -1$. Since HI is odd, this calculates HI explicitly for all values of x.

Since there is an isomorphism mapping any vertex to any other vertex, the degree one Fourier coefficients of HI are all equal, and we might as well calculate the coefficient corresponding to x_1. Now

$$\widehat{HI}(1) = \mathbf{E}[HI(X)X_1] = \frac{\mathbf{E}[HI(X)|X_1 = 1] - \mathbf{E}[HI(X)|X_1 = -1]}{2}$$

Since HI is odd,

$$\mathbf{E}[HI(X)|X_1 = -1] = -\mathbf{E}[HI(-X)|X_1 = -1] = -\mathbf{E}[HI(X)|X_1 = 1]$$

hence $\widehat{HI}(1) = \mathbf{E}[HI(X)|X_1 = 1]$. If we let Y be the random variable denoting the number of $x_i = 1$, for $i \neq 1$, then

$$\begin{aligned}\mathbf{E}[HI(X)|X_1 = 1] &= (1/32)\sum_{k=0}^{5}\binom{5}{k}\mathbf{E}[HI(X)|X_1 = 1, Y = k] \\ &= (1/32)\left(\binom{5}{0} + \binom{5}{1} - \binom{5}{3} - \binom{5}{4} + \binom{5}{5}\right) \\ &= (1/32)(1 + 5 - 10 - 5 + 1) = -1/4\end{aligned}$$

since $\mathbf{E}[HI(X)|X_1 = 1, Y = 2] = (1/10)(5-5) = 0$. To simplify the calculation of coefficients of degree, note that we have graph isomorphisms mapping any triangle to any other triangle, hence the Fourier coefficients of degree 3 corresponding to a triangle are all equal. If Y is the number of $x_i = 1$, for $i \neq 1,2,3$, then by applying the same technique as when we were calculating

$\mathbf{E}[HI(X)|X_1 = 1]$, *we find*

$$\begin{aligned}
\mathbf{E}[HI(X)|X_1 = +1, X_2 = +1, X_3 = +1] &= (1/8)(1-3-3+1) = -1/2 \\
\mathbf{E}[HI(X)|X_1 = +1, X_2 = +1, X_3 = -1] &= (1/8)(1+1-2-3-1) = -1/2 \\
\mathbf{E}[HI(X)|X_1 = +1, X_2 = -1, X_3 = -1] &= -\mathbf{E}[HI(X)|X_1 = +1, X_2 = +1, X_3 = -1] \\
&= 1/2 \\
\mathbf{E}[HI(X)|X_1 = -1, X_2 = -1, X_3 = -1] &= -\mathbf{E}[HI(X)|X_1 = +1, X_2 = +1, X_3 = +1] \\
&= 1/2
\end{aligned}$$

We can then interpolate these results to find

$$\widehat{f}(\{1,2,3\}) = (-1/2 + 3/2 + 3/2 - 1/2)(1/8) = 1/4$$

Now the sum of the squares of the Fourier coefficients we have calculated are $10(1/4)^2 + 6(1/4)^2 = 1$, *so that all other Fourier coefficients are zero. Thus*

$$f(x) = 1/4(e_\Delta - e_1)$$

When e_1 *is the first symmetric polynomial (the sum of all monomials of degree one), and* e_Δ *is the sum of all monomials which correspond to triples of vertices in the hemiicosohedron which form triangles.*

Viewing Boolean functions in the domain $\mathbf{F}_2^n$, we also have an expansion

$$f(x) = \sum a_S x^S$$

for functions $f : \mathbf{F}_2^n \to \mathbf{R}$, where $x^S = \prod_{i \in S} x_i$ is now just the characteristic function, equal to one when all x_i are equal to one. This is very different than the measures of parity on $\{-1,1\}^n$. The expansion is obtained by induction, noting that for functions $f : \mathbf{F}_2 \to \mathbf{R}$,

$$f(x) = f(0) + x[f(1) - f(0)]$$

and that if $f : \mathbf{F}_2^{n+1} \to \mathbf{R}$, then by induction there are $a_S, b_S \in \mathbf{R}$ such that

$$f(x,0) = \sum a_S x^S \qquad f(x,1) = \sum b_S x^S$$

and then

$$f(x,y) = f(x,0) + y[f(x,1) - f(x,0)] = \sum a_S x^S + \sum (b_S - a_S) y x^S$$

so we have an expansion in products. To see that the expansion is unique, consider the linear operator

$$(D_i f)(x) = f(x^{i\to 1}) - f(x^{i\to 0})$$

(This is slightly different to the D_i operators on $\{-1,1\}^n$ that we will analyze later). Then

$$D_i x^S = \begin{cases} x^{S-i} & i \in S \\ 0 & i \notin S \end{cases}$$

So by linearity, if $f(x) = \sum_S a_S x^S$, then $D_i f = \sum_{i\in S} a_S x^{S-i}$. Now if the expansion was not unique, then we could find a_S not all equal to zero with $f(x) = \sum a_S x^S = 0$. If we pick $T = \{i_1,\ldots,i_n\}$ with maximal cardinality, subject to the constraint that $a_T \neq 0$, then $(D_{i_1} D_{i_2} \ldots D_{i_n} f)(x) = a_T$, and if $f = 0$ then $a_T = D_{i_1} \ldots D_{i_n} f = 0$, contradicting the fact that a_T was nonzero. Thus the expansion is unique. Similarily, we can expand arbitrary functions $\mathbf{F}_2^n \to \mathbf{F}_2$ as sums of subsets of characteristic functions x^S.

The expansions of functions on $\mathbf{F}_2^n$ are an interesting phenomenon, but aren't too useful to the study of harmonic analysis. Firstly, the x^S do not form an orthonormal basis for the class of real-valued functions, which makes them less useful. They do diagonalize the multiplication operators $(\sigma_y f)(x_1,\ldots,x_n) = f(x_1 y_1,\ldots,x_n y_n)$, for $y \in \mathbf{F}_2^n$, but these operators are not studied too much – they just annihilate certain coordinates of the domain of f. The technique can be used to obtain Fourier coefficients, because we have an expansion

$$x^S = \frac{1}{2^{|S|}} \sum_{T\subset S} (-1)^T \chi_T(x)$$

hence if $f : \mathbf{F}_2^n \to \mathbf{R}$ is expanded as

$$f(x) = \sum_S a_S x^S \qquad f(x) = \sum_S b_S \chi_S(x)$$

Then

$$b_T = (-1)^{|T|} \sum_{T\subset S} \frac{a_S}{2^{|S|}}$$

In particular, we find that the degrees of both expansions are the same.

1.2 Fourier Analysis and Probability Theory

Many of the interesting results of Boolean Fourier analysis occur in the context of statistical problems over discrete domains. Thus it is of interest to interpret the main tools and results of probability theory in the context of Boolean analysis. Unless otherwise stated, the distributions of all results will be uniform. This means the probability theory rephrases combinatorial results, but it is still a useful source of intuition and elegant notation.

We can define an inner product on Boolean functions by letting

$$\langle f, g \rangle = \frac{1}{2^n} \sum f(x)g(x) = \mathbf{E}[f(X)g(X)]$$

The characters on $\{-1,1\}^n$ form an orthonormal basis to this inner product, and it therefore follows that for any Boolean function f,

$$\widehat{f}(S) = \mathbf{E}[f(X)X^S]$$

hence the Fourier coefficients of f correspond to the average value of f relative to the parity of S. For instance, a Boolean-valued function $f : \{-1,1\}^n \to \{-1,1\}$ is *unbiased*, that is, it takes values 1 and -1 with equal probability, if and only if $\widehat{f}(\emptyset) = 0$.

The standard Hilbert space results give rise to probabilistic interpretations. Parseval's equality means

$$\mathbf{E}[f(X)^2] = \sum \widehat{f}(S)^2$$

which implies that

$$\mathbf{V}[f(X)] = \mathbf{E}[f(X)^2] - \mathbf{E}[f(X)]^2 = \sum_{S \neq \emptyset} \widehat{f}(S)^2$$

$$\mathrm{Cov}(f(X), g(X)) = \sum_{S \neq \emptyset} \widehat{f}(S)\widehat{g}(S)$$

Hence expectations, variances, and covariances are directly related to the Fourier coefficients of the Boolean function in question.

If f and g are *Boolean-valued* maps, then we can write

$$\mathbf{E}[f(X)g(X)] = \mathbf{P}(f(X) = g(X)) - \mathbf{P}(f(X) \neq g(X)) = 2\mathbf{P}(f(X) = g(X)) - 1$$

Note that $\mathbf{P}(f(X) = g(X))$ differs from the Hamming distance of f and g by a constant. We define this value to be the **relative Hamming distance** $d(f,g)$. Since $f^2 = g^2 = 1$, $\|f\|_2^2 = 1$, and this implies that

$$\begin{aligned}\mathbf{V}(f) &= \mathbf{E}[f(X)^2] - \mathbf{E}[f(X)]^2 \\ &= 1 - (\mathbf{P}(f(X) = 1) - \mathbf{P}(f(X) = -1))^2 \\ &= 4d(f,1)d(f,-1)\end{aligned}$$

Thus the variance of Boolean valued functions is very closely related to the degree to which the function is constant.

Lemma 1.1. *If* $\varepsilon = \min[d(f,1), d(f,-1)]$, *then* $2\varepsilon \leqslant \mathbf{V}(f) \leqslant 4\varepsilon$.

Proof. We are effectively proving that for any $x \in [0,1]$,

$$2\min(x, 1-x) \leqslant 4x(1-x) \leqslant 4\min(x, 1-x)$$

Since $4x(1-x) = 4\max(x,1-x)\min(x,1-x)$, we divide by $4\min(x,1-x)$ and restate the inequality as $1/2 \leqslant \max(x,1-x) \leqslant 1$, which is obvious. □

This shows that if we have a sequence of functions $f_n : \{-1,1\}^n \to \{-1,1\}$, then $\varepsilon_n \sim x_n$ holds if and only if $\mathbf{V}(f) \sim x_n$. The minimum hamming distance to a constant value is asymptotically the same as the variation of the function. In general domains we intuitively view $\mathbf{V}(f)$ as a measure of how constant the function f is, but for functions $\{-1,1\}$ we have a precise, quantitative estimate.

Example. *Consider a function f chosen uniformly randomly from the set of all Boolean functions on $\{-1,1\}^n$. That is, for any function $g : \{-1,1\}^n \to \{-1,1\}$, the probability distribution gives $\mathbf{P}(f = g) = 1/2^{(2^n)}$. Since each Boolean function has a Fourier expansion, f has a Boolean expansion*

$$f(x) = \sum \widehat{f}(S) x^S$$

where each $\widehat{f}(S)$ is now a real-valued random variable. Now because we have a bijection amongst the set of all Boolean functions, mapping g to $-g$, we find

$$\begin{aligned}\mathbf{E}[\widehat{f}(S)] &= \frac{1}{2^{2^n}} \sum_g \widehat{g}(S) = \frac{1}{2^{2^n}} \sum \widehat{(-g)}(S) \\ &= -\frac{1}{2^{2^n}} \sum \widehat{g}(S) = -\mathbf{E}[\widehat{f}(S)]\end{aligned}$$

Hence $\mathbf{E}[\widehat{f}(S)] = 0$. *What's more, given that* $X \neq Y$, *the probability that* $f(X) = f(Y)$ *is independent of* X *and* Y, *once we know* $X \neq Y$, *hence*

$$\begin{aligned}
\mathbf{V}[\widehat{f}(S)] = \mathbf{E}[\widehat{f}(S)^2] &= \mathbf{E}\left[\mathbf{E}\left(f(X)X^S\right)^2\right] \\
&= \mathbf{E}[f(X)f(Y)(XY)^S] \\
&= \mathbf{P}(X = Y) + \mathbf{P}(X \neq Y)\mathbf{E}[f(X)f(Y)(XY)^S | X \neq Y] \\
&= \frac{1}{2^n} + \left(1 - \frac{1}{2^n}\right)(2\mathbf{P}(f(X) = f(Y)|X \neq Y) - 1)\mathbf{E}((XY)^S | X \neq Y)
\end{aligned}$$

For any particular $x \neq y$, $\mathbf{P}[f(x) = f(y)] = 1/2$, *because* f *is chosen uniformly from all functions, and for any function* g, *we have the function* g', *such that* $g'(z) = g(z)$ *except when* $z = x$, *where* $g'(z) = -g(x)$, *and the association is a bijection. This implies that* $\mathbf{P}(f(X) = f(Y)|X \neq Y) = 1/2$, *hence the variation satisfies* $\mathbf{V}[\widehat{f}(S)] = 1/2^n$. *Thus functions on average have no Fourier coefficient on each* S, *and most functions have very small Fourier coefficient.*

Since a function $f : \{-1,1\}^n \to \{-1,1\}$ always has square-norm 1, because $f^2 = 1$, Parsevel's theorem implies that $\sum \widehat{f}(S)^2 = 1$ if we sum over all the Fourier coeffients, so that a boolean-valued function on n variables gives rise to a probability distribution over $[n]$. We call this the spectral sample of f, and denote the distribution by $\mathcal{S}_f$.

Often times, the Fourier coeffients of sets over some particular cardinality are more than enough to determine some result. We define the weight of a function $f : \{-1,1\}^n \to \mathbf{R}$ of degree k to be

$$\mathbf{W}^k(f) = \sum_{|S|=k} \widehat{f}^2(S)$$

If f is boolean-valued, then we can also define $\mathbf{W}^k(f) = \mathbf{P}(|S| = k)$ where S is a set randomly drawn from the spectral sample of f. $\mathbf{W}^k(f)$ is also the square norm of the function

$$(P^k f)(x) = \sum_{|S|=k} \widehat{f}(S)x^S$$

which can be seen as a certain truncation estimate of f.

1.3 Testing Linearity

A function $f : \{-1,1\}^n \to \{-1,1\}$ is linear if $f(xy) = f(x)f(y)$ for all $x, y \in \{-1,1\}^n$, or equivalently, if there is $S \subset [n]$ such that $f(x) = x^S$. Given a general function f, there are effectively only two ways to explicitly check that the function is linear, we either check that $f(xy) = f(x)f(y)$ for all choices of the arguments x and y, or check that $f(x) = x^S$ holds for some choice of S, over all choices of x. Even assuming that f can be evaluated in constant time, both methods take exponential time in n to compute. This is guaranteed, because the minimal description length of f is always exponential in n, and if an algorithm determining linearity does not check the entire description of a linear function f, we can modify f to a non-linear function on inputs that the linearity tester doesn't look at, at the algorithm will not be able to distinguish between these two inputs. The problem of testing linearity is a scenario in a family of problesm in the field of **property testing**, which attempts to design efficient algorithms to determine whether a particular boolean-valued function satisfies a certain property.

We might not be able to come up with a polynomial time algorithm to verify linearity, but we can make headway by considering the possiblity of coming up with a randomized algorithm which can verify linearity with high probability. The likely solution to the problem, given some function f, would to perform the linearity test for $f(XY) = f(X)f(Y)$ for a certain set of randomly chosen inputs X and Y. If f is non-linear, then $f(XY) \neq f(X)f(Y)$ with positive probability, and if we find this particular input we can guarantee the function is non-linear. If f is linear, the test always passes. We shall find that the probability that the test fails directly relates to how similar a function is to a linear function.

The simplest version of linearity testing runs using one random query as a test – it just takes one pair of inputs (X, Y), and tests the linearity of this function against these inputs. This is known as the Blum-Luby-Rosenfeld algorithm, or BLR for short. It turns out that the sucess of this method is directly related to how similar a function is to a linear character. There are two candidates to define what it means to be 'approximately linear'.

- $f(xy) = f(x)f(y)$ for a large majority of inputs x and y. Essentially, this means exactly that $f(XY) = f(X)f(Y))$ with high probability, so

the BLR test is accepted with high probability on input f.

- There is a linear functional g such that $f(x) = g(x)$ for a large number of inputs x. This means exactly that $d(f,g)$ is small.

The two notions are certainly related, but a direct relation seems difficult to quantify. One way is easy to bound. If g is linear, and $d(f,g) = \varepsilon$, then a standard union bound inequality guarantees that

$$\begin{aligned}\mathbf{P}[f(X)f(Y) = f(XY)] &\geqslant \mathbf{P}[f(X) = g(X), f(Y) = g(Y), f(XY) = g(XY)] \\ &\geqslant 1 - 3\varepsilon\end{aligned}$$

and we shall find that even tighter bounds hold. But just because $f(xy) = f(x)f(y)$ for a large number of x, y doesn't seem to imply that f is close to any *particular* linear function. For a subset $\mathcal{X}$ of Boolean-valued functions, we shall let $d(f,\mathcal{X}) = \inf_{g\in\mathcal{X}} d(f,g)$. If $\mathcal{C}$ is the set of characters, then we can state our problem as analyzing the relation between how the BLR test interacts with f, and the value of $d(f,\mathcal{C})$.

Our analysis of the BLR test essentially shows that there is a very close relation between these two properties for the case of Boolean functions. Since we are applying Hamming distances in measuring how linear a function is, we can guess that the Harmonic analysis of boolean functions will come in handy.

Theorem 1.2. *For any function $f : \{-1,1\}^n \to \{-1,1\}$, if $d(f,\mathcal{C}) > \varepsilon$, then* $\mathbf{P}$*(BLR algorithm rejects f) $> \varepsilon$.*

Proof. Note that

$$\mathbf{I}[f(XY) \neq f(X)f(Y)] = \frac{1 - f(X)f(Y)f(XY)}{2}$$

Hence

$$\begin{aligned}
\mathbf{P}[\text{BLR rejects } f] &= \mathbf{E}\left[\frac{1 - f(X)f(Y)f(XY)}{2}\right] \\
&= \frac{1}{2} - \frac{1}{2}\mathbf{E}_X[f(X)\mathbf{E}_Y[f(Y)f(XY)]] \\
&= \frac{1}{2} - \frac{1}{2}\mathbf{E}_X[f(X)(f * f)(X)] \\
&= \frac{1}{2} - \frac{1}{2}\sum \widehat{f}(S)^3 \\
&\geqslant \frac{1}{2} - \frac{1}{2}\max \widehat{f}(T)\sum \widehat{f}(S)^2 \\
&= \frac{1}{2} - \frac{1}{2}\max \widehat{f}(T)
\end{aligned}$$

Now for each S, $\widehat{f}(S) = 1 - 2d(f, x^S)$, so if $d(f, x^S) > \varepsilon$ for all S, we find $\widehat{f}(S) < 1 - 2\varepsilon$, hence the probability that the BLR test rejects f is greater than

$$\mathbf{P}[\text{BLR rejects } f] > \frac{1}{2} - \frac{1}{2}(1 - 2\varepsilon) = \varepsilon$$

and this completes the proof. □

The converse of this result is that if the BLR test accepts f with probability $1 - \varepsilon$, then f is ε close to being linear, and so we have shown that $f(XY) = f(X)f(Y)$ holds with high probability if and only if f is ε close to being linear. Note that the BLR algorithm, with only three evaluations of the function f, can determine with high accuracy that f is not linear, if the function is far away from being non-linear to begin with, or, if we are wrong, then f is very similar to a linear function in the first case. Yet given f, we cannot use this algorithm to determine the linear function x^S which f is similar to. Of course, for most x, $f(x) = x^S$, but we do not know for which x this occurs. However, we cannot use this estimate to obtain accurate estimates in expectation for a fixed x, in the sense that there is no measure of the likelihood of the estimate being equal to the actual value. The problem is that x is a fixed quantity, rather than varying over many values of the function, and is therefore our test is easy to break. Note, however, that $y^S = x^S(xy)^S$, and if we let x be a random quantity, then $(xy)^S$ and x^S will likely be equal to $f(xy)$ and $f(y)$ with a probability that is feasible to determine.

Theorem 1.3. *If $d(f, x^S) \geqslant \varepsilon$, then for any x,*

$$\mathbf{P}[f(xY)f(Y) = x^S] \geqslant 1 - 2\varepsilon$$

Proof. We just apply a union bound to conclude

$$\mathbf{P}(f(xY)f(Y) = x^S) \geqslant \mathbf{P}(f(xY) = x^S Y^S, f(Y) = Y^S) \geqslant 1 - 2\varepsilon$$

and in this circumstance, we find $f(yX)f(X) = y^S$. □

Thus linear functions are *locally correctable*, in that we can correct small modifications to linear functions with high probability. This turns out to have vast repurcussions in the theory of property testing and complexity theory, which we will discuss later. One reason why these methods work is that the linear functions on $\mathbf{F}_2^n$ form a very discrete set. For $S \neq T$,

$$d(x^S, x^T) = \mathbf{P}(x^S \neq x^T) = \mathbf{P}(x^{S\Delta T} = -1) = 1/2$$

We find characters are separated by a dimension-independant quantity. If $d(f, x^T) \leqslant \varepsilon$ for some T, then $d(f, x^S) \geqslant 1/2 - \varepsilon$ for $S \neq T$, and therefore it is difficult to splice together two linear functions so that $f(XY) = f(X)f(Y)$ holds with high probability without f being close to a particular linear function.

The inequality in our analysis of the BLR test is tight, provided that all Fourier coefficients are reasonably equal to one another. Analyzing the gap in the inequality, we find

$$\begin{aligned}\max \widehat{f}(T) \sum \widehat{f}(S)^2 - \sum \widehat{f}(S)^3 &= \sum \widehat{f}(S)^2 [\max \widehat{f}(T) - \widehat{f}(S)] \\ &\leqslant \max \widehat{f}(T) - \min \widehat{f}(S)\end{aligned}$$

If all Fourier coefficients are equal, then the inequality is an equality. Though this isn't strictly the spectra of a *Boolean-valued* function (it's the spectra of the indicator function $\mathbf{I}(x = 1)$), we can find Boolean functions f whose Fourier coefficients are arbitrarily close to one another. For instance, the difference between the maximum and minimum coefficients of the inner product functions $f(x, y) = (-1)^{\langle x, y \rangle}$ on $\mathbf{F}_2^n$ is 2^{1-n}. Thus we have found the best dimension independant constant that holds over all functions f. However, if one of the $\widehat{f}(S)$ is much larger than the others, so f is very close to x^S, then we can obtain a better bound, and thus a much tighter analysis of the BLR test.

Theorem 1.4. *If $d(f, x^S) = \varepsilon$, then the BLR test rejects f with probability at least $3\varepsilon - 10\varepsilon^2 + 8\varepsilon^3$.*

Proof. Note that

$$\widehat{f}(S) = \mathbf{E}[f(X)X^S] = 1 - 2d(f, x^S) = 1 - 2\varepsilon$$

and also for $T \neq S$,

$$\widehat{f}(T) = 1 - 2d(f, x^T) \leqslant 1 - 2(d(f, x^S) - d(x^T, x^S)) = 2\varepsilon$$

Hence

$$\begin{aligned}
\mathbf{P}(\text{BLR rejects } f) &= \frac{1}{2} - \frac{1}{2}\sum_{T \neq S} \widehat{f}(T)^3 - \frac{1}{2}\widehat{f}(S)^3 \\
&\geqslant \frac{1}{2} - \varepsilon \sum_{T \neq S} \widehat{f}(T)^2 - \frac{1}{2}\widehat{f}(S)^3 \\
&= \frac{1}{2} - \varepsilon(1 - \widehat{f}(S)^2) - \frac{1}{2}\widehat{f}(S)^3 \\
&= \frac{1}{2} - \varepsilon(1 - (1 - 2\varepsilon)^2) - \frac{1}{2}(1 - 2\varepsilon)^3 \\
&= 3\varepsilon - 10\varepsilon^2 + 8\varepsilon^3
\end{aligned}$$

The bound here is much tighter than the other calculations. Indeed the difference in the inequality is

$$\sum_{T \neq S} \varepsilon \widehat{f}(T)^2 - (1/2)\widehat{f}(T)^3 = \sum_{T \neq S} \widehat{f}(T)^2[\varepsilon - \widehat{f}(T)/2] \leqslant \max(\varepsilon - \widehat{f}(T)^2/2) \leqslant \varepsilon$$

So for small ε the bound is tight. □

The converse of this statement is that if the BLR test accepts f with probability greater than $1 - 3\varepsilon + O(\varepsilon^2)$, then f is ε close to being linear. Thus for small ε the bound on similarity to a linear function is tighter than we initially calculated. This makes sense because, when f is very close to a linear function, it seems intutitive that it is difficult for $f(xy) = f(x)f(y)$ *not* to hold.

This is the best upper bound we can obtain, up to a first order. For each n, consider the function $g_n : \mathbf{F}_2^n \to \mathbf{F}_2$, with $g_n(x) = \mathbf{I}(x = 1)$. Let us count all instances when g_n is rejected by the BLR test. There are three possibilities where $g_n(x + y) \neq g_n(x) + g_n(y)$:

- If $x = 1$ and $y \neq 1$.
- If $x \neq 1$ and $y = 1$.
- If $x \neq 1$ and $y \neq 1$, and $x + y = 1$.

There are $2^n - 1$ instances of the first and second rejection, and $2^n - 2$ instances of the third. Thus

$$\mathbf{P}(\text{BLR rejects } g_n) = \frac{2(2^n - 1) + (2^n - 2)}{4^n} = \frac{3}{2^n} - \frac{4}{4^n}$$

and g_n is $1/2^n$ close to 0. If there was a lower bound of the form $C\varepsilon - O(\varepsilon^2)$, for $C > 3$, and if we write $1/2^n = \varepsilon$, there for any $0 < \delta < C$, such that for big enough n we find $3\varepsilon - 4\varepsilon^2 \geqslant (3 + \delta)\varepsilon$, hence $-4\varepsilon \geqslant \delta$, which is impossible since we can let ε tend to zero as $n \to \infty$.

1.4 The Walsh-Fourier Transform

Historically, the study of the Fourier transform on $\mathbf{F}_2^n$ emerged from a novel analysis of the Fourier expansions of $L^2[0,1]$. In this section we consider this development for completeness. Consider the product group $\mathbf{F}_2^\infty = \prod \mathbf{F}_2$, which is a compact abelian group. We have the standard projections $i_n : \mathbf{F}_2^n \to \mathbf{F}_2^\infty$, defined by $i_n(x_1, \dots, x_n) = (x_1, \dots, x_n, 0, 0, \dots)$. In addition, we have the projections $\pi_n(x_1, x_2, \dots) = (x_1, \dots, x_n)$. Given a character f on $\mathbf{F}_2^\infty$, each $f \circ i_n$ is a character on $\mathbf{F}_2^n$, and can thus be written χ_{S_n} for some $S_n \subset [n]$. The sets S_n defined must be increasing in n, and for $n \leqslant m$, satisfy $S_m \cap [n] = S_n$. For any $x = (x_1, x_2, \dots)$ in $\mathbf{F}_2^\infty$, the sequence $\pi_n(x)$ converges to x, and hence

$$f(x) = \lim \pi_n(x) = \lim x^{S_n}$$

If $S = \lim S_n$ is not a bounded subset of the integers, then the limit above can be allowed to oscillate infinitely between -1 and 1, hence f cannot be a *continuous* character on $\mathbf{F}_2^\infty$. We therefore find that characters on $\mathbf{F}_2^\infty$ correspond to finite subset of positive integers, and topologically is essentially the inductive limit of the character groups $(\mathbf{F}_2^n)^*$.

An important link between probability theory and real analysis is that the binary expansion of real numbers on $[0,1]$,

$$x = \sum_{m=1}^{\infty} \frac{x_m}{2^m}$$

can be seen as defining a measure preserving map between $\mathbf{F}_2^\infty$ and $[0,1]$, where

$$(x_1, x_2, \dots) \mapsto \sum_{m=1}^{\infty} \frac{x_m}{2^m}$$

which is injective almost everywhere. Thus we can find a uniform distribution over $[0,1]$ by taking the binary expansion obtained by flipping a fair coin infinitely many times. Important to our work is that the harmonic analysis of $\mathbf{F}_2^\infty$ gives rise to an expansion theory on $[0,1]$. In particular, we find that the characters χ_S, which form an orthonormal basis for $L^2(\mathbf{F}_2^\infty)$ correspond to an orthonormal basis of functions on $L^2[0,1]$ of the form

$$w^S(x) = \chi_S(x_1, x_2, \dots) = \prod_{i \in S} (-1)^{x_i} = \prod_{i \in S} r_i(x)$$

where $r_i(x) = (-1)^{x_i}$ is the i'th Rademacher function. Because integers have unique binary expansions, we may reindex w^S as w_n for a unique non-negative integer n, as was done classically, and this family of functions are called the Walsh functions. Thus every function has a unique expansion

$$f(x) = \sum_{n=0}^{\infty} a_n w_n(x) = \sum_{\substack{S \subset [n] \\ \#S < \infty}} \widehat{f}(S) w^S(x)$$

and this is essentially where the beginnings of the study of Boolean expansions began, by Walsh in the 1920s. These expansions have very good L^p truncations, but they are less used in modern mathematics because the functions w_n are not smooth, and as such are not useful in the study of partial differential equations.

For the purposes of discrete Boolean analysis, we shall be interested in this system in order to come up with a computationally efficient way of computing the Walsh-Fourier expansion of a boolean valued function $f : \mathbf{F}_2^n \to \mathbf{R}$. For an integer k, define k_n to be the coeffient in the binary expansion

$$k = \sum k_n 2^n$$

First, note that there is a basis $\{X_k\}$ of functions from $\mathbf{F}_2^n \to \mathbf{R}$ defined by

$$X_k = \mathbf{I}(x_1 = k_1, \dots, x_n = k_n)$$

and a basis $\{Y_k\}$ of elements of functions from $\mathbf{F}_2^*$ to $\mathbf{R}$ defined by

$$Y_k = \chi_{\{i:k_i=1\}}$$

Since the Walsh transform $\mathcal{W}$ is effectively a map from $\mathbf{R}^{\mathbf{F}_2^n}$ to $\mathbf{R}^{(\mathbf{F}_2^n)^*}$, these bases induce a matrix representation W of the transform. We have

$$\mathcal{W}(X_l) = \frac{1}{2^n}\sum_S (-1)^{\sum_{m\in S} l_m}\chi_S = \frac{1}{2^n}\sum_k (-1)^{\sum l_i k_i} Y_k$$

Hence if we define the matrices

$$H^1 = \begin{pmatrix} 1 \end{pmatrix}$$

$$H^{n+1} = \begin{pmatrix} H^n & H^n \\ H^n & -H^n \end{pmatrix}$$

Then $W = H_n/2^n$, because, as can be proved by induction, $H^n_{ab} = (-1)^{\sum a_i b_i}$. By a divide and conquer approach, if we write a vector in $\mathbf{R}^{\mathbf{F}_2^n}$ as (v, w), where v and w split the space in half according to the basis defined above, then

$$H_{n+1}(v, w) = (H_n v + H_n w, H_n v - H_n w)$$

Using this approach, if t_n is the maximum number of additions and subtractions required to calculate the Walsh transform a n-dimensional Boolean function, then $t_{n+1} \leqslant 2t_n + 2n$, with $t_1 = 0$, hence

$$t_n \leqslant 2^{n+1}\sum_{k=2}^{n} k\left(\frac{1}{2}\right)^k \leqslant n2^{n+1}\sum_{k=2}^{n}\left(\frac{1}{2}\right)^k \leqslant n2^n$$

so the calculation is possible in $n2^n$ calculations, which looks bad, but the algorithm is polynomial in the size of the input, because an element of $\mathbf{R}^{\mathbf{F}_2^n}$ takes $m = 2^n$ numbers to specify, so the algorithm is $m \lg m$. This is essentially the fastest way to compute the Fourier coefficients of an arbitrary boolean function.

Chapter 2

Social Choice Functions

We now interpret boolean functions in a particular non-abstract scenario in the theory of social choice. Here a Boolean function $f : \{-1,1\}^n \to \{-1,1\}$ is employed as a device in a two-candidate election. A group of n people choose between two candidates, labelled -1 and 1, which are fed into the function f as a voting rule to determine the overall outcome of the vote. In this situation it is natural to find voting rules with various properties relavent to promoting a fair election, and these properties also have an interest in the general context of the harmonic analysis of Boolean functions.

Example. *The standard Boolean function used as a voting rule is the majority function $Maj_n : \{-1,1\}^n \to \{-1,1\}$, which outputs the candidate who got the majority of votes. This is always well defined with an odd number of voters, but there are issues with ties when an even number of voters is given. A function is called a majority function if it always chooses the candidate with the majority of votes where this step is well defined.*

Example. *The voting rule $And_n : \{-1,1\}^n \to \{-1,1\}$ chooses candidate 1 unless every other person votes for candidate -1. Similarily, the voting rule $Or_n : \{-1,1\}^n \to \{-1,1\}$ votes for candidate -1 unless everyone else votes against him.*

Example. *The dictator rule $\chi_i : \{-1,1\}^n \to \{-1,1\}$ defined by $\chi_i(x) = x_i$ places the power of decision making in a single person.*

Example. *All of these functions can be quantified as a version of the weight majority, or linear threshold function, those functions f which can be defined,*

for some $a_0, \ldots, a_n \in \mathbf{R}$, as

$$f(x) = sgn(a_0 + a_1 x_1 + \cdots + a_n x_n)$$

Thus each voter has a certain voting power, specified by the a_i, and there is an additional bias a_0 which pushes the vote in a certain candidate's favour.

Example. *In the United States, a recursive majority voting rule is used. A simple version of the rule is defined on n^d bits by*

$$Maj_n^{\otimes d}(x_1, \ldots, x_d) = Maj_n(Maj_n^{\otimes(d-1)}(x_1), Maj_n^{\otimes(d-1)}(x_n))$$

where each $x_i \in \{-1,1\}^{n^{d-1}}$. Thus we subdivide voters into certain regions, determine the majority over this region, and then take the majority over the accumulated choices.

Example. *The tribes function divides voters into tribes, and the outcome of the vote holds if and only if one tribe is unanimously in favour of the vote. The width w, size s tribe voting rule is defined by*

$$Tribes_{ws} : \{-1,1\}^{sw} \to \{-1,1\}$$

$$Tribes_{ws}(x_1, \ldots, x_s) = Or_s(And_w(x_1), \ldots, And_w(x_s))$$

where each $x_i \in \{-1,1\}^w$. The number of ways we can screw up each individual election is $(2^w - 1)$, so that $\mathbf{P}(Tribes_{ws} = 1) = (2^w - 1)^s / 2^{sw} = (1 - 2^{-w})^s$, and so $\mathbf{E}[Tribes_{ws}] = 2(1 - 2^{-w})^s - 1$. If we control the size of w and s, we can therefore get an unbiased education.

Any boolean function is a voting rule on two candidates, so the study of voting functions is really just a language for talking about general properties of boolean functions. However, it certainly motivates the development of certain contexts which will become very useful in the future. For instance, important properties of Boolean functions become desirable properties of voting rules. In particular,

- A voting rule f is **monotone** if $x \leqslant y$ implies $f(x) \leqslant f(y)$.
- A voting rule is **odd** if $f(-x) = -f(x)$.
- A rule is **unanimous** if $f(1) = 1$, and $f(-1) = -1$.

- A voting rule is **symmetric** if $f(x^\pi) = f(x)$, where $\pi \in S_n$ acts on $\{-1,1\}^n$ by permuting coordinates.
- A voting rule is **transitive symmetric** if, for any index i, there is a permutation π taking i to any other index j, such that $f(x^\pi) = f(x)$.

There is only a single function which satisfies all of these properties for odd n, the majority function Maj_n. This constitutes May's theorem.

Theorem 2.1. *The majority function is the only monotone, odd, unanimous, symmetric, transitive symmetric function in odd dimension. In even dimensions, such a function does not exist.*

Proof. Suppose $f : \{-1,1\}^n \to \{-1,1\}$ is a function which is monotone and symmetric. Then $f(x) = g(\#\{i : x^i = 1\})$ for some monotone function $g : \{0,\dots,n\} \to \{-1,1\}$. The monoticity implies that there is N for which

$$g(n) = \begin{cases} -1 & n < N \\ 1 & n \geqslant N \end{cases}$$

Note that if f is an odd function, then

$$\mathbf{E}[f(X)] = \mathbf{E}[f(-X)] = -\mathbf{E}[f(X)]$$

hence $\mathbf{E}[f(X)] = 0$, so f takes $+1$ and -1 with equal probability. But also

$$0 = \mathbf{E}[f(X)] = \frac{1}{2^n} \sum_{k=0}^{n} \binom{n}{k} g(k)$$

Hence

$$\sum_{k=0}^{N-1} \binom{n}{k} = \sum_{k=N}^{n} \binom{n}{k}$$

Since the left side strictly increases as a function of N, and the right side strictly decreases, if a N exists which satisfies this equality it is unique. If n is odd, then we can pick $N = (n+1)/2$, because the identity

$$\binom{n}{k} = \binom{n}{n-k}$$

which implies coefficients can be paired up. If n is even, the same pairing technique shows that such a choose of N is impossible. □

2.1 Influence

Given a voting rule $f : \{-1,1\}^n \to \{-1,1\}$, we may wish to quantify the power of each voter in changing the outcome of the vote. If we choose a particular voter i and we fix all inputs but the choice of the i'th voter, we say that the voter i is **pivotal** if his choice affects the outcome of the election. That is, given $x \in \{-1,1\}^n$, we define $x^{\oplus i} \in \{-1,1\}^n$ to be the input obtained by flipping the i'th bit. Then i is pivotal on x exactly when $f(x^{\oplus i}) \neq f(x^i)$. It is important to note that even if a voting rule is symmetric, it does not imply that each person has equal power over the results of the election, once other votes have been fixed (For instance, in a majority voitng system with 3 votes for candidate A and 4 votes for candidate B, the voters for B have more power because their choice, considered apart from the others seriously impacts the result of the particular election).

It is natural to define the *overall power* of a voter i on a certain voting rule f as the likelihood that the voter is pivotal on the election. We call this the **influence** of the voter, denoted $\mathrm{Inf}_i(f)$. To be completely accurate, we would have to determine the probability that each particular choice of votes occurs (If there is a high chance that one candidate is going to one, we should expect each voter to individually have little power, whereas if the competition is tight, each voter should have much more power). We assume that all voters choose outcomes independently, and uniformly randomly (this is the **impartial culture** assumption), so that that the influence takes the form

$$\mathrm{Inf}_i(f) = \mathbf{P}(f(X) \neq f(X^{\oplus i}))$$

Thus the influence measures the probability that index i is pivotal uniformly across all elements of the domain.

The impartial culture assumption is not only for simplicity, in the face of no other objective choice of distribution, but also because the formula then has combinatorial interest. If we colour a node on the Hamming cube based on the output of the function f, then the influence of an index i is the fraction of coordinates whose color changes when we move along the i'th dimension of the cube. Similarily, we can also consider it as the fraction of edges along the i'th dimension upon which f changes.

Example. *The dictator function χ_i satisfies $Inf_i(\chi_i) = 1$, and $Inf_j(\chi_i) = 0$ otherwise. If f is an arbitrary voting rule for which $Inf_i(f) = 0$, then f completely*

ignores index i, we can actually specify f as a function of the remaining $n-1$ variables.

Example. *The i'th index is pivotal for And_n only at the points 1 and $1^{\oplus i}$, so that $Inf_i(And_n) = 2/2^n = 2^{1-n}$.*

Example. *To calculate the influence of Maj_n, we note that if $Maj_n(x) = -1$, and $Maj_n(x^{\oplus i}) = 1$, then there are $(n+1)/2$ indices j such that $x^j = -1$, and i is one of these indices. Once i is fixed, the total possible choices of indices in which these circumstances hold is*

$$\binom{n-1}{\frac{n-1}{2}}$$

hence if we also consider x for which $Maj_n(x) = 1$, and $Maj_n(x^{\oplus i}) = -1$, then

$$Inf_i(Maj_n) = 2\binom{n-1}{\frac{n-1}{2}}/2^n = \binom{n-1}{\frac{n-1}{2}}2^{1-n}$$

Applying Stirling's approximation, we can compute the approximation

$$Inf_i(Maj_n) = \sqrt{\frac{2}{n\pi}} + O(n^{-3/2})$$

We will soon show this is about the best influence we can find for a monotonic, symmetric voting rule.

To connect the influence of a boolean function to it's Fourier expansion, we must come up with an analytic expression for the influence. Consider the i'th partial derivative operator

$$(D_i f)(x) = \frac{f(x^{i\to 1}) - f(x^{i\to -1})}{2}$$

In this equation, we see a slight relation to differentation of real-valued functions, and the analogy is completed when we see how D_i acts on the polynomial representation of f. Indeed,

$$D_i x^S = \begin{cases} x^{S-i} & : i \in S \\ 0 & : i \notin S \end{cases}$$

Hence by linearity,

$$(D_i f)(x) = \sum_{i \in S} \widehat{f}(S) x^{S - \{i\}}$$

For Boolean-valued functions f, $D_i f$ is intimately connected to the influence of f, because

$$(D_i f)(x) = \begin{cases} \pm 1 & f(x^i) \neq f(x^{\oplus i}) \\ 0 & f(x^i) = f(x^{\oplus i}) \end{cases}$$

Hence $(D_i f)^2(x)$ is the 0-1 indicator of whether i is pivotal at x, and so

$$\mathrm{Inf}_i(f) = \mathbf{E}[(D_i f)^2] = \|D_i f\|_2^2 = \sum_{i \in S} \widehat{f}(S)^2$$

Defining the i'th **Laplacian** operator $L_i = x_i D_i$, we find

$$(L_i f)(x) = \sum_{i \in S} \widehat{f}(S) x^S$$

Hence L_i is a projection operator, satisfying $\mathrm{Inf}_i(f) = \langle L_i f, L_i f \rangle = \langle f, L_i f \rangle$. Thus we have found a quadratic representation of $\mathrm{Inf}_i(f)$. This will have widespread influences throughout the theory. In particular, we can now already see that an index i is nonessential if and only if $\widehat{f}(S) = 0$ for all sets S such that $i \in S$.

We shall dwell on the Laplacian for slightly longer than the other operators, since we will find it's generalization occuring in future discussion. As a projection operator, it has a complementary projection

$$(E_i f)(x) = \frac{f(x^{i \to 1}) + f(x^{i \to -1})}{2}$$

which can be considered the expected value of f, if we fix all coordinates except for the i'th index, which takes a uniformly random value. Thus for any boolean function f we have

$$f = E_i f + x_i D_i f$$

Both $E_i f$ and $D_i f$ do not depend on the i'th coordinate of f, which is useful for carrying out inductions on the dimension of a boolean function's

domain. We also note that we have a representation of the Laplacian as a difference

$$(L_i f)(x) = \frac{f(x) - f(x^{\oplus i})}{2}$$

so that the Laplacian acts very similarily to D_i.

Though we use the fact that $(D_i f)^2 = |D_i f|$ to obtain a quadratic equation for $D_i f$, the definition of the influence as the absolute value $|D_i f|$ as the definition of influence still has its uses, especially when obtaining L^1 inequalities for the influence. Indeed, we can establish the bound

$$\mathrm{Inf}_i(f) = \mathbf{E}|D_i f| \geqslant |\mathbf{E}(D_i f)| = |\widehat{f}(i)|$$

This inequality only becomes an equality when f is always increasing in the i'th direction, or always decreasing – that is, if $D_i f \geqslant 0$ or $D_i f \leqslant 0$, and if $D_i f \geqslant 0$ we in fact have $\mathrm{Inf}_i(f) = \widehat{f}(i)$, since $\widehat{f}(i)$ is non-negative. In particular, if f is monotone, then we have $\mathrm{Inf}_i(f) = \widehat{f}(i)$ for all i, which is very useful. As a first application of this fact, we show that in a monotone, transitive symmetric voting system, all voters have small influence.

Theorem 2.2. *Given a monotone, transitive symmetric $f : \{-1,1\}^n \to \{-1,1\}$,*

$$Inf_i(f) \leqslant \frac{1}{\sqrt{n}}$$

Proof. Applying Parsevel's inequality, we find

$$1 = \|f\|_2^2 = \sum \widehat{f}(S)^2 \geqslant \sum \widehat{f}(i)^2 = n\widehat{f}(i)^2$$

Hence $\widehat{f}(i) \leqslant 1/\sqrt{n}$. But by monotonicity,

$$\mathrm{Inf}_i(f) = \widehat{f}(i)$$

Putting these equations together gives us the inequality. □

It might be possible to make this inequality tighter. First, note that the inequality is only tight if $\mathbf{W}^1(f) = 1$, in which case f is a dictator or its negation, and thus not transitive symmetric unless $n = 1$. Thus the inequality is always slightly loose. However, I can't think of a way to tighten this inequality, or a better inequality right now.

The second is that there are very few Boolean valued functions with Fourier coefficients bounded by degree one.

Lemma 2.3. *If $f : \{-1,1\}^n \to \{-1,1\}$ has $W^{\leqslant 1}(f) = 1$, then f is either constant, or $\pm x^i$ for some $i \in [n]$.*

Proof. Write $f(x) = a + \sum b_j x_j$. Then $(D_i f)(x) = b_i$. Because f is Boolean valued,

$$(D_i f)(x) = \frac{f(x^{i\to 1}) - f(x^{i\to -1})}{2} \in \{0, \pm 1\}$$

Hence $b_i = 0$ or $b_i = \pm 1$. If all b_i are equal to zero, then f is constant. Otherwise, there is b_i with $b_i^2 = 1$, and then all other coefficients must vanish, and so $f(x) = x_i$ or $f(x) = -x_i$. □

The **total influence** of a boolean function $f : \{-1,1\}^n \to \{-1,1\}$, denoted $\mathbf{I}(f)$, is the sum of the influences $\mathrm{Inf}_i(f)$ over each coordinate. It is not a measure of how powerful each voter is, but instead how chaotic the system is with respect to how any of the voters change their coordinates. For instance, constant functions have total influence zero, where no vote change effects the outcome in any way, whereas the function $x^{[n]}$ has total influence n, since every change in a voters choices flips the outcome of the vote. For monotone transitive symmetric voting rules, we already know the total influence is bounded by $\sqrt{n}$, and this is asymptotically obtained by the majority function.

Example. *The total influence of χ_S is S, because all the indices of S are pivotal for all inputs. The total influence is maximized over all functions by the parity function $x^{[n]}$, which is always pivotal for all inputs. The total influence of And_n and Or_n is $n2^{1-n}$, rather small, whereas Maj_n has total influence*

$$\sqrt{2n/\pi} + O\left(\sqrt{1/n}\right)$$

which is the maximum total influence of any monotone transitive symmetric function up to a scalar multiple.

Example. *The tribes function*

As a sum of quadratic forms, the total influence is also a quadratic form, but cannot be represented by a projection. Considering

$$\mathbf{I}(f) = \sum_i \mathrm{Inf}_i(f) = \sum_i \langle f, L_i f\rangle = \left\langle f, \left(\sum L_i\right) f\right\rangle$$

we see that the total influence is represented as a quadratic form over the **Laplacian** operator $L = \sum L_i$. Note that

$$(Lf)(x) = \sum_{i=1}^{n} \sum_{i \in S} \widehat{f}(S) x^S = \sum |S| \widehat{f}(S) x^S$$

so the Laplacian amplifies the coefficients of f corresponding to sets of large weight. This makes sense, because x^S changes on more inputs when S contains more indices, hence the function should have higher influence. This implies that

$$\mathbf{I}(f) = \sum |S| \widehat{f}(S)^2 = \sum k \mathbf{W}^k(f)$$

for Boolean-valued functions, the total influence is the expected cardinality of $\mathcal{S}$, where $\mathcal{S}$ is a random set distributed according to the spectral distribution induced from f.

There is a probabilistic interpretation of the total influence, which makes it interesting to the theory of voting systems. Define $\mathrm{sens}_f(x)$ to be the number of pivotal indices for f at x. Then $\mathbf{I}(f) = \mathbf{E}[\mathrm{sens}_f(X)]$. Indeed, we find

$$(Lf)(x) = \sum_i \frac{f(x) - f(x^{\oplus i})}{2} = \frac{n}{2}[f(x) - \mathbf{E}[f(x^{\oplus i})]]$$

where the expectation is taken where i is random, uniformly distributed over all indices $[n]$. We have

$$\mathbf{E}[f(x^{\oplus i})] = \frac{n - \mathrm{sens}_f(x)}{n} f(x) + \frac{\mathrm{sens}_f(x)}{n}(-f(x)) = f(x) \frac{n - 2\mathrm{sens}_f(x)}{n}$$

Hence $(Lf)(x) = f(x)\mathrm{sens}_f(x)$.

There is also a combinatorial interpretation of the total influence. Each vertex has n edges, and if we consider the probability distribution which first picks a point on the cube uniformly, and then an edge uniformly, then the resulting distribution will be the uniform distribution on edges. Since the expected number of **boundary edges** (those (x, y) for which $f(x) \neq f(y)$) eminating from a vertex is $\mathbf{I}(f)$, the chance that the edge we pick will be a boundary edge is $\mathbf{I}(f)/n$.

Since $\mathbf{V}(f) = \sum_{k\neq 0} \mathbf{W}^k(f)$, and $\mathbf{I}(f) = \sum_{k\neq 0} k\mathbf{W}^k(f)$, it is fairly trivial to obtain the Poincare inequality $\mathbf{V}(f) \leqslant \mathbf{I}(f)$, which gives a strict inequality unless $\mathbf{W}^{>2} f = 0$. If the low degree coefficients of the function vanish, then we can give an even sharper inequality – if $\mathbf{W}^{<k}(f) = 0$, then $\mathbf{V}(f) \leqslant \mathbf{I}(f)/k$. Conversely, $\mathrm{Inf}_i(f) \leqslant \mathbf{V}(f) \leqslant \mathbf{I}(f)$, so the variance is likely close to the total influence when the influence is concentrated on a single voter. For Boolean valued functions, the only such functions are the singular characters $\pm x^i$ and the constant functions. Geometrically, the poincare inequality gives an edge expansion bound for functions on the Hamming cube. Given a subset A of the n dimensional Hamming cube with m points, and if $\alpha = m/2^n$, then the 'characteristic function' $I_f : \{-1,1\}^n \to \{-1,1\}$ has variation $4\alpha(1-\alpha)$, and therefore the fraction of edges in the cube which form a boundary edge is lower bounded by $4\alpha(1-\alpha)/n$. In particular, for the set A corresponding to $x^{[n]}$, where $\alpha = 1/2$, we see this bound is tight, as for $\alpha = 0$ and $\alpha = 1$. However, for other values of α much better bounds can be obtained.

Lemma 2.4. *If $f : \{-1,1\}^n \to \{-1,1\}$ is an unbiased Boolean function, then there is i with $Inf_i(f) \geqslant 2/n - 4/n^2$.*

Proof. Let I be the index which maximizes $\mathrm{Inf}_I(f)$ over all choices of indicies. First, note that $\mathrm{Inf}_I(f) \geqslant 1/n$, for if $\mathrm{Inf}_i(f) < 1/n$ holds for all i, then we find $\mathbf{I}(f) < 1 = \mathbf{V}(f)$, which is impossible. Now, because f is unbiased,

$$\mathbf{I}(f) \geqslant \mathbf{W}^1(f) + 2(1 - \mathbf{W}^1(f)) = 2 - \mathbf{W}^1(f)$$

Now we use the fact that $|\widehat{f}(i)| \leqslant \mathrm{Inf}_i(f)$ to conclude that

$$\mathbf{W}^1(f) = \sum \widehat{f}(i)^2 \leqslant n\mathrm{Inf}_I(f)^2$$

hence $n\mathrm{Inf}_I(f) \geqslant \mathbf{I}(f) \geqslant 2 - n\mathrm{Inf}_I(f)^2$. Rearranging, we can calculate that $\mathrm{Inf}_I(f) \geqslant 2/n - \mathrm{Inf}_I(f)^2$, hence

$$\mathrm{Inf}_I(f) \geqslant \min_{x\in[n^{-1},1]} \max(x, 2n^{-1} - x^2) = \frac{\sqrt{1+8n^{-1}} - 1}{2} \geqslant (2/n) - (4/n^2)$$

Thus unbiased Boolean functions give some index an influence of at least $2/n + O(1/n^2)$. □

Returning to our discussion of voting systems, an additional property to consider is the expected number of voters who agree with the outcome of the vote.

Theorem 2.5. *Let f be a voting rule for a 2-candidate election. If $w(x)$ is the number of x^i which agree with the voting choice, then*

$$\mathbf{E}(w) = \frac{n}{2} + \frac{1}{2}\sum \widehat{f}(i)$$

which is maximized by the majority functions.

Proof. We have

$$w(x) = \sum_{i=1}^{n} \frac{1 + f(x)x^i}{2}$$

Hence

$$\mathbf{E}(w(X)) = \frac{n}{2} + \frac{1}{2}\sum_{i=1}^{n} \mathbf{E}[f(x)x^i] = \frac{n}{2} + \frac{1}{2}\sum_{i=1}^{n} \widehat{f}(i)$$

We write

$$\sum_{i=1}^{n} \widehat{f}(i) = \mathbf{E}(f(X)(X^1 + X^2 + \cdots + X^n)) \leqslant \mathbf{E}(|X^1 + X^2 \cdots + X^n|)$$

and this inequality turns into an equality precisely when f has the property that $f(X) = \operatorname{sgn}(\sum X^i)$, whenever $\sum X^i \neq 0$. □

Example. *Consider the function f choosen randomly from all Boolean valued functions on $\{-1,1\}^n$. We calculate*

$$\mathbf{E}[Inf_i(f)] = \mathbf{E}_X[\mathbf{P}(f(X) \neq f(X^{\oplus i}))] = \mathbf{E}_X[1/2] = 1/2$$

Hence $\mathbf{E}[\mathbf{I}(f)] = n/2$, so for an average function half of the voters agree with the outcome.

2.2 Noise and Stability

In realistic voting systems, the inputs to voting systems are rarely the most reliable. Some voters don't even show up to voting booths to announce

their choice of candidate, and errors in the recording of data mean that the input might not even be accurate to the choices of voters. It is important for a voting rule to remain stable under these pertubations, so that the outcome of the modifed vote is likely to be the same as the outcome of the original vote, if we had perfect knowledge of all voting choices.

So our general rule with which we will define the **stability** of a voting rule f, is to find the probability that $f(x) \neq f(y)$, where y is obtained by a slight pertubation of x. We'll say two Boolean-valued random variables X and Y are ρ-correlated, for some $\rho \in [-1,1]$ if $\mathbf{E}[XY] = \rho$. This is equivalent to

$$\mathbf{P}(Y = X) = \frac{1+\rho}{2}$$

Given a random variable X, we can generate a random variable Y which is ρ-correlated to X by letting $Y = X$ with probability ρ, and otherwise choosing Y uniformly randomly. We write $Y \sim N_\rho(X)$ is Y is ρ-correlated to X. This is the sense with which we will need ρ-correlation, for ρ-correlation represents a certain value being randomly perturbed by a small quantity. Similarily, if X and Y are multidimensional Boolean-valued functions, we say X is ρ-correlated to Y if the coordinates of X and Y are independant, and each coordinate in X is ρ-correlated to the corresponding coordinate in Y. We can now define the stability with respect to ρ as

$$\mathrm{Stab}_\rho(f) = \mathop{\mathbf{E}}_{Y \sim N_\rho(X)}[f(X)f(Y)]$$

where X is chosen uniformly. We of course find

$$\mathrm{Stab}_\rho(f) = 2\left(\mathop{\mathbf{P}}_{Y \sim N_\rho(X)} f(X) = f(Y)\right) - 1$$

Thus if the voting system is modified by some small random quantity ρ, the chance that this will affect the outcome of the vote is $[1 + \mathrm{Stab}_\rho(f)]/2$.

An alternate measure of the sensitivity of a function, which may be easier to reason with, is the probability that the vote is disrupted if we purposefully reverse each coordinate of X independently with a small probability δ, forming the random variable Y. Thus $\mathbf{P}(Y = X) = 1 - \delta$, hence Y is $1 - 2\delta$-correlated to X. We define the **noise sensitivity** based on this correlation to be

$$\mathbf{NS}_\delta(f) = \mathbf{P}[f(X) \neq f(Y)]$$

Since Y is $1-2\delta$ correlated with X, and by our previous formula, we find

$$\mathbf{NS}_\delta(f) = \frac{1 - \mathrm{Stab}_{1-2\delta}(f)}{2}$$

So that there is a linear correlation between the two measures of stability.

Example. *The constant functions ± 1 have stability 1, since there is no chance of changing the value of the functions. The dictator functions x^i satisfy $\mathbf{NS}_\delta(x^i) = \delta$, hence*

$$Stab_\rho(x^i) = 1 - 2\mathbf{NS}_{\frac{1-\rho}{2}}(x^i) = \rho$$

More generally, the stability of x^S is

$$Stab_\rho(x^S) = \mathbf{E}(X^S Y^S) = \prod_{i \in S} \mathbf{E}(X^i Y^i) = \prod_{i \in S} \left(\left(\frac{1+\rho}{2} \right) - \left(\frac{1-\rho}{2} \right) \right) = \rho^{|S|}$$

The noise stability of the majority functions has no convenient formula, but it does tend to a nice limit at the number of voters increases ad infinum. Later on, we will prove that for $\rho \in [-1, 1]$,

$$\lim_{\substack{n \to \infty \\ n \text{ odd}}} Stab_\rho(Maj_n) = \frac{2}{\pi} \arcsin(\rho) = 1 - \frac{2}{\pi} \arccos(\rho)$$

Hence for $\delta \in [0, 1]$,

$$\lim_{\substack{n \to \infty \\ n \text{ odd}}} \mathbf{NS}_\delta[Maj_n] = (1/\pi) \arccos(1 - 2\delta) = \frac{2\sqrt{\delta}}{\pi} + O(\delta^{3/2})$$

Hence the noise stability curves sharply near small and large values of ρ.

That noise stability is tightly connected to the Fourier coefficients of the function is one of the most powerful tools in Boolean harmonic analysis. As with the influence, we introduce an operator to provide us with a connection to the coefficients. We introduce the **noise operator** with noise parameter ρ (also called the **Bonami-Beckner operator** in the literature) as

$$(T_\rho f)(x) = \mathop{\mathbf{E}}_{X \sim N_\rho(x)} f(X)$$

which 'smoothes out' the function f by a certain parameter ρ. T_ρ is obviously linear, and

$$(T_\rho x^S) = \mathbf{E}[X^S] = \prod_{i \in S} \mathbf{E}[X^i] = \prod_{i \in S} \left(\left(\frac{1+\rho}{2} \right) x^i - \left(\frac{1-\rho}{2} \right) x^i \right) = \rho^{|S|} x^S$$

Hence the noise operator transforms the Fourier coeffients as

$$(T_\rho f)(x) = \sum \rho^{|S|} \widehat{f}(S) x^S$$

The connection between the noise operator and the stability is that

$$\begin{aligned} \mathrm{Stab}_\rho(f) &= \mathbf{E}[f(X)f(Y)] = \mathbf{E}[f(X)\mathbf{E}[f(Y)|X]] \\ &= \mathbf{E}[f(X)(T_\rho f)(X)] = \langle f, T_\rho f \rangle \end{aligned}$$

Thus stability is a quadratic form, and so we find

$$\mathrm{Stab}_\rho(f) = \sum_S \rho^{|S|} \widehat{f}(S)^2 = \sum_k \rho^k \mathbf{W}^k(f)$$

Correspondingly, we have

$$\begin{aligned} \mathbf{NS}_\delta(f) &= \frac{1 - \mathrm{stab}_{1-2\delta}(f)}{2} \\ &= \frac{1 - \mathbf{W}^0(f)}{2} - \sum_{k \neq 0} \frac{(1-2\delta)^k}{2} \mathbf{W}^k(f) \\ &= \frac{1}{2} \sum (1 - (1-2\delta)^k) \mathbf{W}^k(f) \end{aligned}$$

Hence $\mathbf{NS}_\delta$ can also be represented as a quadratic form.

That all the operators that we have found can be considered as a quadratic form should not be taken as a hint that all interesting operators in Boolean analysis are quadratic, just that the ones easiest to understand happen to be. And these operators are even more easy to understand because they are diagonalized by the characters x^S, which all happen to be Eigenvalues. And if that wasn't enough, all the eigenvalues are positive, which implies that the functionals are convex – given a functional F with $F(\sum a_S x^S) =$

$\sum b_S a_S^2$, where $b_S \geqslant 0$, and given f and g, using the convexity of x^2 we find

$$\begin{aligned} F(\lambda f + (1-\lambda)g) &= \sum_S b_S \left[\lambda \widehat{f}(S) + (1-\lambda)\widehat{g}(S)\right]^2 \\ &\leqslant \sum_S b_S \left[\lambda \widehat{f}(S)^2 + (1-\lambda)\widehat{g}(S)^2\right] \\ &\leqslant \lambda F(f) + (1-\lambda)F(g) \end{aligned}$$

Thus these operators are incredibly applicable to understanding.

Theorem 2.6. *For any $\rho, \mu \in [-1,1]$, $T_\rho \circ T_\mu = T_{\rho\mu}$.*

Proof. If X is μ correlated to the constant $x \in \{-1,1\}$, and Y is ρ correlated to X, then Y is $\rho\mu$ correlated to x, since

$$\begin{aligned} \mathbf{P}(Y = x) &= \mathbf{P}(X = x)\mathbf{P}(Y = X) + \mathbf{P}(X \neq x)\mathbf{P}(Y \neq X) \\ &= \left(\frac{1+\mu}{2}\right)\left(\frac{1+\rho}{2}\right) + \left(\frac{1-\mu}{2}\right)\left(\frac{1-\rho}{2}\right) = \frac{1+\rho\mu}{2} \end{aligned}$$

Hence we have shown $(T_\rho \circ T_\mu)(f)(x) = (T_{\rho\mu} f)(x)$ for all inputs x. □

Theorem 2.7. *T_ρ is a contraction on $L^q\{-1,1\}^n$ for all $q \geqslant 1$.*

Proof. because by Jensen's inequality, since $x \mapsto x^q$ is convex,

$$\begin{aligned} \|T_\rho f\|_q^q &= \mathbf{E}[(T_\rho f)^q(X)] \\ &= \mathbf{E}[\mathbf{E}_{Y \sim N_\rho(X)}[f(Y)]^q] \\ &\leqslant \mathbf{E}[\mathbf{E}_{Y \sim N_\rho(X)}[f^q(Y)]] \end{aligned}$$

If X is chosen uniformly randomly, and $Y \sim N_\rho(X)$, then Y is also chosen uniformly randomly, since

$$\begin{aligned} \mathbf{P}(Y = 1) &= \mathbf{P}(Y = X)\mathbf{P}(X = 1) + \mathbf{P}(Y \neq X)\mathbf{P}(X = -1) \\ &= \frac{\mathbf{P}(Y = X) + \mathbf{P}(Y \neq X)}{2} = \frac{1}{2} \end{aligned}$$

Hence

$$\mathbf{E}[\mathbf{E}_{Y \sim N_\rho(X)}[f^q(Y)]] = \mathbf{E}[f^q(X)] = \|f\|_q^q$$

and we may obtain the inequality by taking q'th roots. □

Using the Fourier representation, we can show that the dictator functions maximize stability among the unbiased Boolean-valued functions.

Theorem 2.8. *For $\rho \in (0,1)$, if f is an unbiased Boolean-valued function, then $Stab_p(f) \leqslant p$, with equality if and only if f is a dictator.*

Proof. We write

$$\mathrm{Stab}_\rho(f) = \sum \rho^{|S|} \widehat{f}(S)^2 \leqslant \rho \sum \widehat{f}(S)^2 = \rho$$

If this inequality is an equality, then $\mathbf{W}^{>1}(f) = 0$, and it then follows that f is either constant or a dictator function, and the constant functions are biased. □

Notice that $\mathrm{Stab}_\rho(f)$ is a polynomial in ρ, with non-negative coefficients. It follows that the stability of a function increases as ρ increases for positive ρ, and $\mathrm{Stab}_0(f) = 1$. What's more, we find that, looking at the coefficients, that

$$\frac{d\mathrm{Stab}_\rho(f)}{d\rho} = \sum_{S \neq \varnothing} |S| \rho^{|S|-1} \widehat{f}(S)^2$$

Hence the derivative at $\rho = 0$ is $\mathbf{W}^1(f)$, and the derivative at $\rho = 1$ is $\mathbf{I}(f)$. We can also use this equation with the mean value theorem to determine a useful bound.

Theorem 2.9. *For any Boolean function f,*

$$|Stab_\rho(f) - Stab_{\rho-\varepsilon}(f)| \leqslant \frac{\varepsilon}{1-\rho} \mathbf{V}(f)$$

Proof. Using the mean value theorem, we can find $\eta \in (\rho - \varepsilon, \rho)$ such that

$$\mathrm{Stab}_\rho(f) - \mathrm{Stab}_{\rho-\varepsilon}(f) = \varepsilon \frac{d\mathrm{Stab}_\eta(f)}{d\eta} = \varepsilon \sum k \eta^{k-1} \mathbf{W}^k(f)$$

Now we use the fact that $k\eta^{k-1} \leqslant 1/(1-\eta)$ for all $0 \leqslant \eta \leqslant 1$, hence

$$\varepsilon \sum k \eta^{k-1} \mathbf{W}^k(f) \leqslant \frac{\varepsilon}{1-\eta} \mathbf{V}(f) \leqslant \frac{\varepsilon}{1-\rho} \mathbf{V}(f)$$

Hence we have a Lipschitz inequality for $\mathrm{Stab}_\rho(f)$ which explodes at the boundary, because the estimate $\sum x^k = 1/(1-x)$ we used to establish this estimate breaks down at $\rho = 1$. □

We can incorporate stability into the concept of influence. For $\rho \in [0,1]$, define the ρ-**stable influence**

$$\mathrm{Inf}_i^{\rho}(f) = \mathrm{Stab}_{\rho}(D_i f) = \sum_{i \in S} \rho^{|S|-1} \widehat{f}(S)^2$$

The **total** p-**stable influence** is

$$\mathbf{I}^{\rho}(f) = \sum \mathrm{Inf}_i^{\rho}(f) = \sum_{i \in S} |S| \rho^{|S|-1} \widehat{f}(S)^2$$

In this form, we see that $\mathbf{I}^{\rho}(f) = d\mathrm{Stab}_{\rho}(f)/d\rho$, so that this ρ-stable influence measures the change in stability of a function with respect to ρ. There isn't too much of a natural interpretation of the ρ-stable influences. Since $D_i f$ measures if f is pivotal at f, one intuitive insight about the ρ-stable influence is that it is a way of smoothening out how pivotal $D_i f$ is; Though $D_i f$ only measures the change in f over the i'th coordinate, the ρ-stable influence now incorporates a slight change in the other coordinates as well. Regardless of their interpretation, they will be technically very useful later on.

Theorem 2.10. *Suppose that a function f satisfies $\mathbf{V}(f) \leqslant 1$. If $0 < \delta, \epsilon < 1$, then the number of indices i for which $Inf_i^{(1-\delta)}(f) \geqslant \varepsilon$ is less than or equal to $1/\delta\varepsilon$.*

Proof. Let J be the set of indices for which $\mathrm{Inf}_i^{(1-\delta)}(f) \geqslant \varepsilon$, so that we must prove $|J| \leqslant 1/\delta\varepsilon$. We obviously have $|J| \leqslant \mathbf{I}^{(1-\delta)}(f)/\varepsilon$, hence we need only prove $\mathbf{I}^{(1-\delta)}(f) \leqslant \frac{1}{\delta}$. Since

$$\begin{aligned}\mathbf{I}^{(1-\delta)}(f) &= \sum |S|(1-\delta)^{|S|-1} \widehat{f}(S)^2 \\ &\leqslant \max(|S|(1-\delta)^{|S|-1}) \sum \widehat{f}(S)^2 = \max(|S| p^{|S|-1})\end{aligned}$$

Hence it suffices to prove that $k\delta(1-\delta)^{k-1} \leqslant 1$ for any $0 < \delta < 1$ and $k \geqslant 1$. Note that for $0 < x < 1$,

$$nx^n \leqslant \sum_{k=0}^{n} x^k \leqslant \sum_{k=0}^{\infty} x^k = 1/(1-x)$$

Our inequality follows by setting $x = 1 - \delta$. □

2.3 Arrow's Theorem

The majority system is perfect for a two-candidate voting system. It is monotone, symmetric, and maximizes the number of voters which agree with each choice. But for three-candidate preferential voting system, we have no information. Given a series of rankings of n candidates, we desire a way to decide upon a winner. In 1785, the French philosopher and mathematician Nicolas de Condorcet suggested breaking down individual voting preferences of voters into pairwise choices over candidates. Once these candidates are considered pairwise, we can then evaluate which candidate is the best over all comparisons. Thus we require a certain voting rule $f : \{-1,1\}^n \to \{-1,1\}$ to consider candidates pairwise, to consider such **Condorcet elections**. In this section, we consider only 3-candidate scenarios.

For instance, consider the Condorcet election over 3 candidates a, b, and c using the majority function as a voting rule. Consider three voters, with preferences $a > b > c$, $a > c > b$, and $b > c > a$. We see that b is favoured more often than c, whereas a is favoured more often than both b and c, hence a is the overall winner of the election. We say a is the **Condorcet winner**. Given a particular voter, we define the xy choice of the voter to be the element of $\{-1,1\}$, where 1 indicates the voter favours x, and -1 favours y. We can identify a particular choice of preference of a voter by consider their ab preference, their bc preference, and their ca preference, which is an element of $\{-1,1\}^3$, and conversely, an element of $\{-1,1\}^3$ gives rise to a unique linear preference, provided that not all of the coordinates of the vector are equal, that is, the vector isn't $\{-1,-1,-1\}$ or $\{1,1,1\}$.

Unfortunately, Condorcet discovered that his election system may not lead to a definite winner. Consider the preferences $c > a > b$, $a > b > c$, and $b > c > a$, using the Condorcet method. Then a is preferred to b, b is preferred to c, and c is preferred to a, so there is no definite winner. In particular, if we consider the winner of each election as a tuple of ab, bc, and ca preferences in $\{-1,1\}^3$, then we can reconstruct a Condorcet winner if and only if the coordinates of the preference vector aren't $\{-1,-1,-1\}$ or $\{1,1,1\}$. Thus the majority rule appears to be deficient in a preferential election.

It was Kenneth Arrow in the 1950s who found that there is always a chance that a Condorcet winner does not occur with *any* voting rule to de-

cide upon a pariwise Condorcet winner, except for a particullary unattractive option: the dictator function. In 2002, Kalai gave a new proof of this theorem using the tools of Fourier analysis. As should be expected from our analysis, it looks at the properties of the 'not all equal' function NAE : $\{-1,1\}^3 \to \{0,1\}$. Instead of looking at a particular element of the domain of the function to determine whether a Condorcet winner is not always possible, Kalai instead computes the *probability* of a Condorcet winner of a voting rule, where the preferences are chosen over all possibilities.

Lemma 2.11. *The probability of a Condorcet winner in a Condorcet election using the pairwise voting rule $f : \{-1,1\}^n \to \{-1,1\}$ is precisely*

$$\frac{3}{4}[1 - Stab_{-1/3}(f)]$$

Proof. Let $x, y, z \in \{-1,1\}^n$ be chosen such that (x_i, y_i, z_i) gives the (ab, bc, ca) preferences of a voter i. The x, y, z are chosen uniformly, in such a way that each (x_i, y_i, z_i) is chosen independently, and uniformly over the 6 tuples which satisfy the not all equal predicate. There is a Condorcet winner if and only if $\mathrm{NAE}(f(x), f(y), f(z)) = 1$, hence

$$\mathbf{P}(f \text{ gives a Condorcet winner}) = \mathbf{E}[\mathrm{NAE}(f(X), f(Y), f(Z))]$$

and since NAE has the Fourier expansion

$$\mathrm{NAE}(x, y, z) = \frac{3 - xy - xz - yz}{4}$$

Hence

$$\mathbf{E}[\mathrm{NAE}(f(X), f(Y), f(Z))] = \frac{3}{4} - \frac{\mathbf{E}(f(X)f(Y)) + \mathbf{E}(f(Y)f(Z)) + \mathbf{E}(f(Z)f(X))}{4}$$

Finally, note that each coordinate of X,Y,Z is independant, and

$$\mathbf{E}[XY] = \mathbf{E}[YZ] = \mathbf{E}[ZX] = 2/6 - 4/6 = -1/3$$

Thus (X,Y), (Y,Z), and (Z,X) are $-1/3$ correlated, hence

$$\mathbf{E}[\mathrm{NAE}(f(X), f(Y), f(Z))] = (3/4)(1 - \mathrm{stab}_{-1/3}(f))$$

and this gives the required formula. □

Example. *Using the majority function Maj_n, we find that the probability of a Condorcet winner tends to*

$$\frac{3}{4}(1-(2/\pi)\sin^{-1}(-1/3)) = \frac{3}{2\pi}\cos^{-1}(-1/3)$$

thus there is approximately a 91.2% *chance of a Condorcet winner occuring. This is Guilbaud's formula.*

Theorem 2.12 (Kalai)**.** *The only voting rule $f : \{-1,1\}^n \to \{-1,1\}$ which always gives a Condorcet winner is the dictator functions x^i, or its negation.*

Proof. If f always gives a Condorcet winner, the chance of a winner is 1, hence rearranging the formula just derived, we find $\mathrm{Stab}_{-1/3}(f) = -1/3$. But then

$$\sum(-1/3)^k \mathbf{W}^k(f) = -1/3$$

Since $(-1/3)^k > -1/3$ for $k > 1$, we find

$$\sum(-1/3)^k \mathbf{W}^k(f) \geqslant -1/3 \sum \mathbf{W}^k(f) = -1/3$$

and this inequality becomes an equality if and only if $\mathbf{W}^{>1}(f) = 0$, which we have verified implies that f is constant, a dictator function, or it's negation. Assuming f is unanimous, we find that f must be the dictator function. □

We might wonder which voting rule gives us the largest chance of a Condorcet winner. It turns out that we can only obtain a slightly more general result than for the majority function.

Lemma 2.13. *If $f : \{-1,1\} \to \{-1,1\}$ has all $\widehat{f}(i)$ equal, then*

$$\mathbf{W}^1(f) \leqslant 2/\pi + O(1/n)$$

Proof. Since $n\widehat{f}(i) = \sum \widehat{f}(i) \leqslant \sqrt{2n/\pi} + O(\sqrt{1/n})$ (the majority function maximizes $\sum \widehat{f}(i)$), we find

$$\mathbf{W}^1(f) = n\widehat{f}(i)^2 = \frac{1}{n}(n\widehat{f}(i))^2 \leqslant 2/\pi + O(1/n)$$

Hence the inequality holds. □

Theorem 2.14. *In a 3-candidate Condorcet election with n voters, the probability of a Condorcet winner is at most* $(7/9) + (2/9)\mathbf{W}^1(f)$

Proof. Given any $f : \{-1,1\}^n \to \{-1,1\}$, the probability of a Condorcet has a Fourier expansion

$$\begin{aligned}\frac{3}{4} - \frac{3}{4}\mathrm{Stab}_{-1/3}(f) &= \frac{3}{4} - \frac{3}{4}\sum(-1/3)^k \mathbf{W}^k(f) \\ &= \left(\frac{3}{4} - \mathbf{E}[f]\right) + \frac{\mathbf{W}^1(f)}{4} - \frac{\mathbf{W}^2(f)}{12} + \frac{\mathbf{W}^3(f)}{36} - \dots \\ &\leqslant (3/4) + \frac{\mathbf{W}^1(f)}{4} + \frac{1}{36}\left(\sum_{k=2}^{n} \mathbf{W}^k(f)\right) \\ &\leqslant (3/4) + \frac{\mathbf{W}^1(f)}{4} + \frac{1 - \mathbf{W}^1(f)}{36} = (7/9) + (2/9)\mathbf{W}^1(f)\end{aligned}$$

□

Combining these two results, we can conclude that in any voting system with all $\widehat{f}(i)$ equal (which is certainly true even in a transitive symmetric system), the chance of a Condorcet winner is

$$(7/9) + (2/9)\mathbf{W}^1(f) \leqslant (7/9) + (4/9\pi) + O(1/n)$$

And this is approximately 91.9%. Later on, we will be able to show that this bound holds given a much weaker hypothesis, and what's more, we will show that the majority function actually maximizes the probability.

The advantage of Kalai's approach is that we can use it to obtain a much more robust result, ala linearity testing, using some more complex theorems of the harmonic analysis.

Theorem 2.15. *If the probability of a Condorcet election of* $1 - \varepsilon$*, then* f *is* $O(\varepsilon)$ *close to being a dictator, or the negation of a dictator.*

Proof. Using the bound we just used, we find $1 - \varepsilon \leqslant (7/9) + (2/9)\mathbf{W}^1(f)$, hence

$$1 - (9/2)\varepsilon \leqslant \mathbf{W}^1(f)$$

and then the result follows from the Friedgut Kalai Naor theorem. □

Theorem 2.16 (Friedgut-Kalai-Naor theorem)**.** *If* f *is a Boolean valued function, and* $\mathbf{W}^1(f) \geqslant 1 - \delta$*, then* f *is* $O(\delta)$ *close to* $\pm x^i$ *for some index* i*.*

The Friedgut-Kalai-Naor requires some sophisticated techniques in Boolean function theory, and we'll prove it in a later chapter.

Chapter 3

Spectral Complexity

In this chapter, we discuss ways we can measure the 'complexity' of a Boolean function, which take the form of various definitions relating to the Fourier coefficients of the function, and the shortest way we can describe the function's output. This discussion has immediate applications to learning theory, where we desire functions which can easily be learned (the simplest functions), and cryptography, where we desire functions which cannot be easily deciphered.

3.1 Spectral Concentration

One way for a function to be 'simple' is for its Fourier coeffients to be concentrated on coefficients of small degree. Call a Boolean function f ε-**concentrated** on degree n if $\mathbf{W}^{\geqslant n}(f) \leqslant \varepsilon$. For Boolean valued functions, this is equivalent to saying that $\mathbf{P}(|S| > n) \leqslant \varepsilon$, where S has the induced spectral distribution. The total influence provides an initial estimate of the spectral concentration.

Theorem 3.1. *A Boolean f is ε-concentrated on degree n, for $n \geqslant \mathbf{I}(f)/\varepsilon$.*

Proof. If we write

$$\mathbf{I}(f) = \sum_k k\mathbf{W}^k(f) \leqslant \varepsilon n$$

then $n\mathbf{W}^{\geqslant n}(f) \leqslant \sum_k k\mathbf{W}^k(f)$, and combining these inequalities completes the proof. For boolean valued functions, this is essentially Markov's inequality applied to the spectral distribution. □

Example. *The tribes function $Tribes_{w,2^w}$ has total influence $O(\log(n))$, hence for any constant ε, the tribes function is ε-concentrated on degree up to $O(\log(n))/\varepsilon$. Since $\log(n)$ grows so slowly, the tribes function is concentrated very strongly on low degree coefficients.*

Another estimate of the spectral concentration is obtained from the noise stability.

Theorem 3.2. *For any Boolean-valued function f, and $0 < \delta \leqslant 1/2$, the Fourier spectrum of f is ε-concentrated on degree $n \geqslant 1/\delta$ for*

$$\varepsilon = \frac{2}{1-e^{-2}}\mathbf{NS}_\delta(f) \leqslant 3\mathbf{NS}_\delta(f)$$

Proof. Using the spectral formula for the noise sensitivity, if S takes on the spectral distribution, then applying the Markov inequality, we find

$$\begin{aligned} 2\mathbf{NS}_\delta(f) &= \mathbf{E}[1-(1-2\delta)^{|S|}] \\ &\geqslant (1-(1-2\delta)^{1/\delta})\mathbf{P}(|S| \geqslant 1/\delta) \\ &\geqslant (1-e^{-2})\mathbf{P}(|S| \geqslant n) \end{aligned}$$

where we used that $1-(1-2\delta)^k$ is a non-negative, non-decreasing function of k, and $(1-2\delta)^{1/\delta} \leqslant e^{-2}$. □

Example. *For δ sufficiently small, and n sufficiently large, the noise sensitivity of Maj_n with parameter δ is bounded by $\sqrt{\delta}$. Hence Maj_n is $3\sqrt{\delta}$ concentrated on degree m for $m \geqslant 1/\delta$.*

We can push ε-concentration all the way down to 0-concentration on degree n, which is the same as saying the Fourier expansion of the function in question has degree less than or equal to n. For these function, we can actually control how many coordinates of the function are useful to the function.

Lemma 3.3. *If f is a real-valued Boolean function with $f \neq 0$, and the degree of the Fourier expansion is less than or equal to m, then $\mathbf{P}(f(X) \neq 0) \geqslant 2^{-m}$.*

Proof. We prove this by induction on the number of coordinates of f. If $f(x) = a$ is a constant function, and $a \neq 0$, then $\mathbf{P}(f(X) \neq 0) = 1 \geqslant 2^{-0}$.

If $f(x) = a + bx$, and $b \neq 0$, then either $f(1)$ or $f(-1)$ are non-zero, so the lemma is satisfied here. In general, if we write

$$f(x) = \sum \widehat{f}(S) x^S$$

and then we define $n-1$ dimensional functions $f_1(x) = f(x, 1)$, and $f_{-1}(x) = f(x, -1)$, then

$$f_1(x) = \sum_{n \notin S} \left(\widehat{f}(S) + \widehat{f}(S \cup \{i\}) \right) x^S \qquad f_{-1}(x) = \sum_{n \notin S} \left(\widehat{f}(S) - \widehat{f}(S \cup \{i\}) \right) x^S$$

Then either f_1 has degree $m-1$, or f_{-1} has degree $m-1$, and it follows by induction that

$$\mathbf{P}(f(X) \neq 0) = \frac{\mathbf{P}(f_1(X) \neq 0) + \mathbf{P}(f_{-1}(X) \neq 0)}{2} \geqslant \frac{1}{2^{(m-1)}} \frac{1}{2} = 2^{-m}$$

and by induction, the bound holds for all functions f. □

Using this lemma, since $\mathrm{Inf}_i(f) = \mathbf{P}((D_i f)(X) \neq 0)$, and if $\deg(f) \leqslant k$, then $\deg(D_i f) \leqslant k-1$, so either $\mathrm{Inf}_i(f) = 0$, or $\mathrm{Inf}_i(f) \geqslant 2^{1-k}$.

Theorem 3.4. *If* $\deg(f) \leqslant k$, *then* f *is a* $k2^{k-1}$ *junta.*

Proof. Because $\mathrm{Inf}_i(f) = 0$ or $\mathrm{Inf}_i(f) \geqslant 2^{1-k}$, the number of coordinates with non-zero influence on f is at most $\mathbf{I}(f)/2^{1-k}$, and those coordinates with non-zero influence are irrelevant to the functions definition. Since

$$\mathbf{I}(f) = \sum_{m \leqslant k} m \mathbf{W}^m(f) \leqslant k(\mathbf{W}^1(f) + \cdots + \mathbf{W}^k(f)) = k$$

hence $\mathbf{I}(f)/2^{1-k} \leqslant k2^{k-1}$, and so f is a $k2^{k-1}$ junta. □

The Friedgut Kalai Naor theorem is essentially a more robust version of this theorem, in the case where $k = 1$. What's more, we shall find that the next section gives us a reason why we can't improve this theorem by much more, because decision trees provide us with a method of constructing $\deg(f) \leqslant k$ functions which can access $k2^{k-1}$ coordinates.

3.2 Decision Trees

Another classification of 'simple' Boolean functions occurs by analyzing the decision trees which represent the function. A **decision tree** for a function $f : \mathbf{F}_2^n \to \mathbf{R}$ is a representation of f by a binary tree in which the internal nodes of the tree are labelled by variables, edges labelled 0 or 1, and leaves by real numbers, such that $f(x)$ can be obtained by starting at the head of the tree, then proceeding down based on the input and the labels of the tree. The **size** of the tree is the number of leaves, and the **depth** is the maximum possible length of a branch from root to leaf. The depth counts the maximum number of questions we need to ask to determine the output of a function f.

Essentially, a decision tree subdivides $\mathbf{F}_2^n$ into cubes upon which the represented function is constant. These cubes can be identified as the set of inputs which follows the same path P, and we shall denote the cube by C_P. It follows that

$$f(x) = \sum_{\text{paths } P \text{ of } T} f(P)\mathbf{I}(x \in C_P)$$

This essentially tells us that low-depth decision trees should have low spectral complexity, because the length of paths bound the degree of the Fourier coefficients. Introducing terminology which will soon become more important, we define the **sparsity** of a function f, denoted sparsity(f), to be the number of elements of its domain which are non-zero. We let the **spectral sparcity** of f be sparsity$(\widehat{f})$, it is the number of subsets $S \subset [n]$ such that $\widehat{f}(S) \neq 0$. Also define f to be ε-**granular** if $f(x)$ is a multiple of ε for any x in the domain.

Theorem 3.5. *If f can be computed by a decision tree of size s and depth k, then*

(1) $\deg(f) \leqslant k$.

(2) $sparsity(\widehat{f}) \leqslant s2^k \leqslant 4^k$

(3) $\|\widehat{f}\|_1 \leqslant s\|f\|_\infty \leqslant 2^k\|f\|_\infty$.

(4) $\widehat{f}$ *is* 2^{-k}*-granular, assuming* f *is integer valued.*

Proof. If a path P has depth m, querying if $x_{i_1} = a_{i_1}, x_{i_2} = a_{i_2}, \ldots, x_{i_m} = a_{i_m}$, then

$$\begin{aligned}\mathbf{I}(x \in C_P) &= \mathbf{I}(x_{i_1} = a_1, \ldots, x_{i_m} = a_m) \\ &= \frac{1}{2^m} \prod_{j=1}^{m} \left(1 + a_j x_{i_j}\right) = \frac{1}{2^m} \sum_{S \subset \{i_1, \ldots, i_m\}} a^S x^S\end{aligned}$$

which we see has a Fourier expansion with degree less than or equal to m. It follows that if each path determining f has degree bounded by k, then the degree of the Fourier coefficients of f are bounded by k, because of the sum formula we calculated above, hence (1) follows. We also see that each path of length m adds at most 2^m Fourier coefficients to the equation, hence (2) follows (also, we see that similar branches share many of the same Fourier coefficients, hence this bound can often be tightened). What's more,

$$f(x) = \sum_{\text{paths } P \text{ of } T} \frac{f(P)}{2^{l(P)}} \sum_{S \subset \{i_1^P, \ldots, i_{l(P)}^P\}} a_P^S x^S$$

Thus

$$\|\widehat{f}\|_1 \leqslant \sum_{\text{paths } P \text{ of } T} \sum_{S \subset \{i_1^P, \ldots, i_{l(P)}^P\}} \frac{|f(P)|}{2^{l(P)}} \leqslant s \|f\|_\infty$$

hence (3) follows. Since $a_P^S \in \{-1, 1\}$, if $f(P) \in \mathbf{Z}$ then $f(P) a_P^S / 2^{l(P)}$ is 2^{-k} granular, hence, summing up, we find all the coefficients of f are. □

Lemma 3.6. *Suppose the Fourier spectrum of $f : \{-1, 1\}^n \to \mathbf{R}$ is ε concentrated on $\mathcal{F}$ and $g : \{-1, 1\}^n \to \mathbf{R}$ satisfies $\|f - g\|_2^2 \leqslant \delta$. Then g is $2(\varepsilon + \delta)$ concentrated on $\mathcal{F}$.*

Proof. TODO: □

Theorem 3.7. *If $f : \{-1, 1\}^n \to \{-1, 1\}$ is computable by a decision tree of size s, and $0 < \varepsilon \leqslant 1$, then the spectrum of f is ε concentrated on degree $m \geqslant \log(s/\varepsilon)$.*

Proof. Let T be the decision tree of size s, and consider the tree T_0 obtained by truncating each branch of length greater than or equal to $\log(s/\varepsilon)$. New leaves created can be chosen arbitrarily. Then the function g obtained from T_0 is ε-close to f, because g only differs from f on the paths of length

$l \geqslant \log(s/\varepsilon)$, and since each query the decision tree makes subdivides the domain in two, changes only $2^{-l} \leqslant 2^{-\log(s/\varepsilon)} = \varepsilon/s$ elements of the domain, and since we only have s paths, we have only changed ε elements of the domain. Since if $m \geqslant \log(s/\varepsilon)$, $\widehat{g}(S)$ vanishes for $|S| \geqslant m$, hence

$$\mathbf{W}^{\geqslant m}(f) \leqslant \|\widehat{g} - \widehat{f}\|_2^2 = \|g - f\|_2^2 \leqslant 4\varepsilon$$

TODO: COME UP WITH TIGHTER BOUND LATER. □

3.3 Computational Learning Theory

Computational learning theory attempts to solve the following problem, motivated from statistics. Given a class of functions $\mathcal{C}$, we attempt to determine an unknown **target function** $f \in \mathcal{C}$ with limited access to that function. We shall assume, of course, that our Boolean-valued functions. The two models of obtaining data to learn the target function are:

- **random examples**, where we draw random samples $(X, f(X))$, where X is uniformly random on the domain.
- **queries**, where we can request a specific value x for any $x \in \{-1, 1\}^n$.

The query method is at least as powerful as the random example method, because we can always randomly choose x ourselves. After a certain number of steps, we are required to choose a **hypothesis function** $h \in \mathcal{C}$. We say that an algorithm **learns** $\mathcal{C}$ **with error** ε if for any $f \in \mathcal{C}$, the algorithm outputs a hypothesis function (not necessarily in $\mathcal{C}$) which is ε-close to f with high probability. The particular accuracy probability baseline is irrelevant, since we can always adjust the baseline by introducing a polynomial number of steps.

As should be intuitive, the more 'simple' the class $\mathcal{C}$ is, the faster it should be to recognize a hypothesis function. If $\mathcal{C}$ is suitably complex, then exponential running time may be necessary. Generalizing our definition of concentration of spectrum, we say f is ε concentrated on $\mathcal{F} \subset 2^{[n]}$ if

$$\sum_{S \notin \mathcal{F}} \widehat{f}(S)^2 \leqslant \varepsilon$$

The main result is that learning a function is equivalent to learning it's Walsh Fourier spectrum, so that functions concentrated on a small portion of the Fourier space are simpler to learn.

Theorem 3.8. *Given access to random samples to a Boolean-valued function, there is a randomized algorithm which take $S \subset [n]$ as input, $0 \leqslant \delta, \varepsilon \leqslant 1/2$, and outputs an estimate $\widetilde{f}(S)$ for $\widehat{f}(S)$ such that*

$$\mathbf{P}(|\widetilde{f}(S) - \widehat{f}(S)| > \varepsilon) \leqslant \delta$$

in $\log(1/\delta) \cdot Poly(n, 1/\varepsilon)$ *time.*

Proof. We have $\widehat{f}(S) = \mathbf{E}[f(X)X^S]$. Given random samples

$$(X_1, f(X_1)), \ldots, (X_m, f(X_m))$$

we can compute $f(X_k)X_k^S$, and then estimate $\mathbf{E}[f(X)X^S]$. A standard application of the Chernoff bound shows that $O(\log(1/\delta)/\varepsilon^2)$ examples are sufficient to obtain an estimate within $\pm\varepsilon$ with probability at least $1 - \delta$. □

Theorem 3.9. *If f is a Boolean-valued function, g is a real-valued Boolean function, and $\|f - g\|_2^2 \leqslant \varepsilon$. Let $h(x) = sgn(g(x))$, with $sgn(0)$ chosen arbitrarily. Then $d(f, h) \leqslant \varepsilon$.*

Proof. Since $|f(x) - g(x)|^2 \geqslant 1$ when $f(x) \neq \operatorname{sgn}(g(x))$,

$$d(f, h) = \mathbf{P}(f(X) \neq h(X)) \leqslant \mathbf{E}[|f(X) - g(X)|^2] = \|f - g\|_2^2$$

Thus we can accurately 'discretize' a real-valued estimate of a Boolean-valued function. □

These theorems allow us to show that estimating Fourier coeficients suffices to estimate the function.

Theorem 3.10. *If an algorithm has random sample access to a Boolean function f, and can identify a family $\mathcal{F}$ upon which f's Fourier spectrum is ε concentrated, then in $poly(|\mathcal{F}|, n, 1/c)$ time, we can with high probability produce a hypothesis function $\varepsilon + c$ close to f.*

Proof. For each $S \in \mathcal{F}$, we produce an estimate $\widetilde{f}(S)$ such that

$$|\widetilde{f}(S) - \widehat{f}(S)| \leqslant \delta$$

except with probability bounded by ρ, in $\log(1/\rho) \cdot \operatorname{poly}(n, 1/\delta)$ time. Applying a union bound, we find all inequalities hold except with probability $\delta|\mathcal{F}|$. Finally, we form the function $g(x) = \sum \tilde{f}(S)x^S$, and then output

$h(x) = \operatorname{sgn}(g(x))$. Now by the last lemma, it suffices to bound the L_2 norm of f and g, and

$$\begin{aligned}\|f-g\|_2^2 &= \sum \left(\widehat{f}(S) - \widehat{g}(S)\right)^2 \\ &= \sum_{S\in\mathcal{F}} \left(\widehat{f}(S) - \tilde{f}(S)\right)^2 + \sum_{S\notin\mathcal{F}} \widehat{f}(S)^2 \\ &= |\mathcal{F}|\delta^2 + \varepsilon\end{aligned}$$

Hence h is $|\mathcal{F}|\delta^2 + \varepsilon$ close to f, and we have computed h in $\log(1/\rho) \cdot \operatorname{poly}(n, 1/\delta)$ time. If we let $\delta = \sqrt{c/|\mathcal{F}|}$, we find h is $c+\varepsilon$ close to f, in time $\log(1/\rho) \cdot \operatorname{poly}(n, |\mathcal{F}|, 1/c)$. □

If we assume that we are learning over a simple concept class to begin with, this theorem essentially provides all the information we need to know in order to learn efficiently over this concept class.

Example. *If our concept class consists of elements in $\mathcal{C}$, which only contains functions of degree $\leqslant k$, then $\mathcal{C}$ is 0-concentrated on the Fourier coefficients of degree k, which contains*

$$|\mathcal{F}| \leqslant \sum_{i=0}^{k} \binom{n}{k} = O(n^k)$$

elements. Thus in $\log(1/\rho) \cdot poly(n^k, 1/\varepsilon)$, we can learn an element of $\mathcal{C}$ with probability $1 - 1/\rho$ up to an accuracy ε.

Example. *If $\mathcal{C}$ consists of elements f with $\mathbf{I}(f) \leqslant t$, then each element is ε concentrated on degree past t/ε, thus we can learn elements of this set in $\log(1/\rho) \cdot poly(n^{t/\varepsilon}, 1/c)$ with error probability ρ, and error $\varepsilon + c$.*

Example. *If $\mathcal{C}$ consists solely of monotone functions, then the total influence of the functions is bounded by $\sqrt{2n/\pi} + O(1/\sqrt{n}) = O(\sqrt{n})$, hence we can learn elements of $\mathcal{C}$ in $\log(1/\rho) \cdot poly(n^{O(\sqrt{n})/\varepsilon}, 1/c)$ with error probability ρ, and error $\varepsilon + c$.*

Example. *If a class $\mathcal{C}$ consists of functions f such that $\mathbf{NS}_\delta(f) \leqslant 3\varepsilon$, for $\delta \in (0, 1/2]$, then f is ε concentrated on degree $1/\delta$, hence $\mathcal{C}$ is learnable with error $\varepsilon + c$ and error probability ρ in time $\log(1/\rho) \cdot poly(n^{1/\delta}, 1/c)$*

Example. *If a class $\mathcal{C}$ consists of functions f which can be represented by a decision tree with size bounded by s, then the spectrum of f is ε-concentrated on degree $\log(s/\varepsilon)$, hence $\mathcal{C}$ is learnable with error $\varepsilon + c$ and error probability ρ in time $\log(1/\rho) \cdot poly(n^{\log(s/\varepsilon)}, 1/c) = \log(1/\rho) \cdot poly(s/\varepsilon, 1/c)$.*

3.4 Walsh-Fourier Analysis Over Vector Spaces

It will be helpful to introduce some notation before attacking the Goldreich Levin theorem. Note that each $\{0,1\}^n$ is a vector space over the field $\{0,1\}$. What's more, we have a map $\langle x, y\rangle = (-1)^{\sum x_i y_i}$ which is 'bilinear', in the sense that

$$\langle x + y, z\rangle = \langle x, z\rangle\langle y, z\rangle \qquad \langle x, y + z\rangle = \langle x, y\rangle\langle x, z\rangle$$

so the map is a homomorphism from $\{0,1\}^n$ to $\{-1,1\}$ in one variable if we fix the other. Each $x \in \{0,1\}^n$ then gives rise to a character x^* defined by $x^*(y) = \langle x, y\rangle$, and $(x+y)^* = x^* y^*$, so the map is a homomorphism. Since the bilinear form $(x, y) \mapsto \sum x_i y_i$ is non-degenerate, the map $x \mapsto x^*$ is injective, and since the character group has the same cardinality as the group, the homomorphism is actually an isomorphism; every character on $\{0,1\}^n$ must be represented by x^* for some x. In fact, if $\chi_S : \{0,1\}^n \to \{-1,1\}$ is a character, then the corresponding x^* is found by letting x be the $\{0,1\}$ indicator function corresponding to S, letting $x_i = 1$ if $i \in S$, and $x_i = 0$ otherwise. Because of this correspondence, we will write $\widehat{f}(x)$ for the corresponding $\widehat{f}(S)$, which can be defined as

$$\widehat{f}(x) = \mathbf{E}[f(X)x^*(X)] = \mathbf{E}[f(X)\langle x, X\rangle]$$

and this is now defined invariant of the original Fourier coefficients.

Given a subspace V of $\{0,1\}^n$, we let $V^\perp$ be the subspace of elements x of $\{0,1\}^n$ such that $\langle x, y\rangle = 1$ for all $y \in V$. By general properties of non-degenerate bilinear forms, $(V^\perp)^\perp = V$, and we can write $\{0,1\}^n = V \oplus V^\perp$, so $V^\perp$ has dimension $n - m$.

Theorem 3.11. *If V is a subspace of $\{0,1\}^n$ of codimension m, then*

$$\mathbf{I}(x \in V) = \frac{1}{2^m} \sum_{y \in V^\perp} y^*$$

Proof. Since $(V^\perp)^\perp = V$, a vector x is in V if and only if $y^*(x) = 1$ for all $y \in V^\perp$. Let $y_1, \ldots, y_m$ form a basis for $V^\perp$. Then

$$\begin{aligned}\mathbf{I}(x \in V) &= \frac{1}{2^m}\prod_{y=1}^{m}\left(y_i^*(x)+1\right) = \frac{1}{2^m}\sum_S\left(\prod_{i\in S} y_i^*(x)\right)\\ &= \frac{1}{2^m}\sum_S\left(\sum_{i\in S} y_i\right)^*(x) = \frac{1}{2^m}\sum_{y\in V^\perp} y^*(x)\end{aligned}$$

□

More generally, if V is an affine subspace of $\{0,1\}^n$, with $V = x + W$ for $x \in V$. We can describe it equivalently as the set of points y such that for any $z \in W^\perp$, $z^*(y) = z^*(x)$. Then if W has codimension m, then

$$\mathbf{I}(y \in V) = \mathbf{I}(y + x \in W) = \frac{1}{2^m}\sum_{z\in W^\perp} z^*(y+x) = \sum_{z\in W^\perp} z^*(x)z^*(y)$$

Thus the indicator has sparsity 2^m, and is 2^{-m} granular.

Given $f : \{-1,1\}^n \to \mathbf{R}$, it will be more natural to think of the function as $f : \{-1,1\}^{[n]} \to \mathbf{R}$, i.e. the function which takes an input $x : [n] \to \{-1,1\}$, and outputs a number $f(x)$. We can consider the Fourier expansion of an arbitrary $f : \{-1,1\}^A \to \mathbf{R}$, where A is an arbitrary index set, and then the characters will take the form x^S for $S \subset A$, and we can consider the Fourier coefficients. Let (J, J^c) be a partition of $[n]$. Then for $z \in \{-1,1\}^J$, we can define $f_{J|z} : \{-1,1\}^{J^c} \to \mathbf{R}$ to be the function obtained by fixing all indices of J with the values z. It follows that if $S \subset J^c$, then

$$\widehat{f_{J|z}}(S) = \sum_{T\subset J}\widehat{f}(S\cup T)z^T$$

Picking z randomly over all choices, we find

$$\mathbf{E}_Z[\widehat{f_{J|Z}}(S)] = \sum_{T\subset J}\widehat{f}(S\cup T)\mathbf{E}[Z^T] = \widehat{f}(S)$$

and

$$\mathbf{E}_Z[\widehat{f_{J|Z}}(S)^2] = \sum_{T\subset J}\widehat{f}(T\cup S)^2$$

Given f and $S \subset J \subset [n]$, let

$$\mathbf{W}^{S|J^c}(f) = \sum_{T \subset J} \widehat{f}(S \cup T)^2$$

which is the weight over irrelevant indices. A crucial tool of Golreich-Levin is that this is the expected value of $\widehat{f_{J|Z}}(S)^2$, so that we can estimate this expected value using the law of large numbers by picking particular values of Z.

Given a Boolean function, it is natural to restrict the function to affine subspaces of $\{0,1\}^n$, for we find that on the subspaces the Fourier coefficients tend to naturally separate. For instance, when a Boolean function is specified by a decision tree, then each path of the decision tree corresponds naturally to a certain subcube of $\{0,1\}^n$, which is an affine subspace. The nice part about restricting functions to certain sub index sets is that we have a canonical base point with which to represent the affine subspace. For an index set $J = \{i_1, \ldots, i_n\}$, $f_{J|z}$ is really the restriction of f to the coset $z + \text{span}(e_{i_1}, \ldots, e_{i_n})$, because by fixing the point z in a coset we have a natural bijection between $z + \text{span}(e_{i_1}, \ldots, e_{i_n})$ and $\text{span}(e_{i_1}, \ldots, e_{i_n})$; the choice of z is natural because it is the only point in $z + \text{span}(e_{i_1}, \ldots, e_{i_n})$.

3.5 Goldreich-Levin Theorem

We have now reduced our problem to identifying a family $\mathcal{F}$ upon which f is concentrated. One such method is provided by the Goldreich-Levin algorithm, which can find $\mathcal{F}$ assuming query access to f. In order for the algorithm to terminate in polynomial time, we require that there is a set $\mathcal{F}$ which exists and is small, yet the algorihtm will find a concentration family $\mathcal{F}$ given arbitrary input.

First, we consider the algorithm's origin in crytography. The goal of this field of study is to find 'encryption functions' f such that if x is a message, then $f(x)$ is easy to compute, but it is very difficult to compute x given $f(x)$. The strength of the encryption f is the difficulty in how it can be inverted. The Goldreich Levin theorem is a tool for building strong encryption schemes from weak schemes. Essentially, this breaks down into finding linear function slightly correlated to some fixed function f, given query access to the function.

The Goldreich Levin theorem assumes query access to a Boolean-valued function $f : \{-1,1\}^n \to \{-1,1\}$, and some fixed $\tau > 0$. Then in time $\text{poly}(n, 1/\tau)$, the algorithm outputs a family $\mathcal{F}$ of subsets of $[n]$ such that

1. If $|\widehat{f}(S)| \geqslant \tau$, $S \in \mathcal{F}$.
2. If $S \in \mathcal{F}$, $|\widehat{f}(S)| \geqslant \tau/2$.

Given a particular S, we know we can compute the estimated Fourier coefficients $\tilde{f}(S)$ to an arbitrary degree of precision with a high degree of accuracy. The problem is that if we did this for all S, then our algorithm wouldn't terminate in polynomial time. The Goldreich-Levin algorithm uses a divide-and-conquer strategy to measure the Fourier weight of the function over various collections of sets.

Theorem 3.12. *For any $S \subset J \subset [n]$, an algorithm with query access to $f : \{-1,1\}^n \to \{-1,1\}$ can compute an estimate of $\mathbf{W}^{S|J^c}(f)$ that is accurate to with $\pm\varepsilon$, except with probability at most δ, in poly$(n, 1/\varepsilon) \cdot \log(1/\delta$ time.*

Proof. We can write

$$\begin{aligned}\mathbf{W}^{S|J^c}(f) = \mathbf{E}_Z[\widehat{f_{J|Z}}(S)^2] &= \mathbf{E}_Z\left[\mathbf{E}_Y\left[f_{J|Z}(Y)Y^S\right]^2\right] \\ &= \mathbf{E}_Z\left[\mathbf{E}_{Y_0,Y_1}\left[f_{J|Z}(Y_0)Y_0^S f_{J|Z}(Y_1)Y_1^s\right]\right]\end{aligned}$$

using queries to f, we can sample the inside of this equation, and a Chernoff bound shows that $O(\log(1/\delta)/\varepsilon^2)$ samples are enough for the estimate to have accuracy ε with confidence $1-\delta$. □

We can now describe the Goldreich Levin algorithm. Initially, all subsets of $[n]$ are put in a single 'bucket'. We then repeat the following process:

- Select any bucket B with 2^m sets in it.
- Split B into two buckets B_1 and B_2 of 2^{m-1} sets.
- Estimate $\sum_{U\in B_1}\widehat{f}(U)^2$ and $\sum_{U\in B_2}\widehat{f}(U)^2$.
- Discard B_1 and B_2 if the weight estimate is less than or equal to $\tau^2/2$.

The algorithm stops when each bucket contains a single item. Note that we never discard a set S with $|\widehat{f}(S)| \geqslant \tau$, because $\tau^2 \geqslant \tau^2/2$. Furthermore, if a bucket containing two elements splits into two buckets containing a single element, then if S is undiscarded in this process, we must have $|\widehat{f}(S)| \geqslant \tau/2$, else $|\widehat{f}(S)|^2 \leqslant \tau^2/4 \leqslant \tau^2/2$. Note that we need only calculate the estimates up to $\pm\tau^2/4$ for the algorithm to be correct, so we have some wriggle room to work with.

Now we need to determine how fast it takes for the algorithm to terminate. Every bucket that isn't discarded has weight $\tau^2/4$, so Parseval's theorem tells us that there is never more than $4/\tau^2$ active buckets. Each bucket can only be split n times, hence the algorithm repeats its main loop $4n/\tau^2$ times. If the buckets can be accurately weighed and maintained in $\mathrm{poly}(n, 1/\tau)$ time, the overall running time will be $\mathrm{poly}(n, 1/\tau)$.

Finally we describe how to weight the buckets. Let

$$B_{k,S} = \{S \cup T : T \subset \{k+1, k+2, \ldots, n\}\}$$

Then $|B_{k,S}| = 2^{n-k}$, the initial bucket is $B_{0,\varnothing}$, and any bucket $B_{k,S}$ splits into $B_{k+1,S}$ and $B_{k+1,S\cup\{k\}}$. The weight of $B_{k,S}$ is $\mathbf{W}^{S|\{k+1,\ldots,n\}}(\mathrm{f})$, and can be measured to accuracy $\pm\tau^2/4$ with confidence $1-\delta$ in $\mathrm{poly}(n, 1/\tau) \cdot \log(1/\delta)$. The algorithm needs at most $8n/\tau^2$ weighings, hence by setting $\delta = \tau^2/(80n)$, we can ensure all weights are accurate with high probability, and the algorithm is $\mathrm{poly}(n, 1/\tau)$.

Chapter 4

Property Testing

We now study the interconnected fields of property testing, probabilistically checkable proofs, and constraint satisfication problems. Our work centers around dictatorship testing, checking if an arbitrary boolean-valued function is a dictator. It turns out that being able to test if a certain function is a dictator is sufficient to test *any* property of boolean-valued functions, provided that the right 'proof' is provided.

4.1 Property Testing

The field of property testing asks a simple problem. Given a collection $\mathcal{C}$ of n-bit Boolean-valued functions, how easy is to determine if an arbitrary boolean functions lies in $\mathcal{C}$ using (not necessarily uniformly) random examples or queries of the function. An **n-query testing algorithm** for $\mathcal{C}$ is an algorithm which randomly chooses elements $x_1, \ldots, x_n \in \{0,1\}^n$, queries $f(x_1), \ldots, f(x_n)$, and decides deterministically whether $f \in \mathcal{C}$. An n-query **local tester** with rejection rate $\lambda > 0$ is a test which always accepts f if $f \in \mathcal{C}$, and if $d(f, \mathcal{C}) > \varepsilon$, then $\mathbf{P}(\text{tester rejects } f) > \lambda\varepsilon$. An alternate statement of this is that if f is accepted with probability greater than $1-\varepsilon$, then $d(f, \mathcal{C}) \leqslant \varepsilon/\lambda$.

Example. *We have seen that the BLR test is a 3-query local tester with rejection rate 1 for the class of linear functions. For large acceptence rates, the tester acts like it has rejection rate* $1/3$.

Example. *A very similar test to the BLR test checks if a function $f : \{-1,1\}^n \to \{-1,1\}$ is affine (that is, if $f = \pm g$ for some linear function g). The class of affine functions can also be describes as the set of functions f such that for any inputs x, y, z, $f(xyz) = f(x)f(y)f(z)$, for certainly all affine functions satisfy this property, and conversely, if a function satisfies this property, then the function $g(x) = f(x)f(1)$ is linear, because*

$$g(xy) = f(xy)f(1) = f(1xy)f(1) = [f(x)f(1)][f(y)f(1)] = g(x)g(y)$$

Thus, given a function $f : \{-1,1\}^n \to \{-1,1\}$, we can consider a 4-query property test which uniformly picks X, Y, and $Z \in \{-1,1\}^n$, and then tests if $f(XYZ) = f(X)f(Y)f(Z)$. The chance of the test passing f is then

$$\begin{aligned}\mathbf{E}\left(\frac{1 + f(XYZ)f(X)f(Y)f(Z)}{2}\right) &= \frac{1}{2} + \frac{1}{2}\mathbf{E}[f(XYZ)f(X)f(Y)f(Z)] \\ &= \frac{1}{2} + \frac{1}{2}\sum \widehat{f}(S)^4\end{aligned}$$

If the test passes with probability greater than $1 - \varepsilon$, then we conclude

$$\sum \widehat{f}(S)^4 > 1 - 2\varepsilon$$

If $d(f, \mathcal{A}) > \varepsilon$, then $d(f, x^S) > \varepsilon$ and $d(f, -x^S) > \varepsilon$ for each linear function x^S, hence $|\widehat{f}(S)| < 1 - 2\varepsilon$, and so

$$1 - 2\varepsilon < \sum \widehat{f}(S)^4 < (1 - 2\varepsilon)^2 = 1 - 4\varepsilon + 4\varepsilon^2$$

Hence $4\varepsilon^2 - 2\varepsilon = 2\varepsilon(2\varepsilon - 1) > 0$. But this implies $\varepsilon > 1/2$, which is impossible, since any function is $1/2$ close to some affine function. Thus we conclude $d(f, \mathcal{A}) \leqslant \varepsilon$, so the affine checker is a rate one, four query local tester.

We could also consider adaptive testing algorithms, which allow us to choose each x_i sequentially after viewing $f(x_1), \ldots, f(x_{i-1})$, or we could consider local testors with more general rejection rates (quadratic or logarithmic instead of linear, for instance), and to allow the number of queries to depend on the error rate. For now, we'll focus on linear local testers.

4.2 Dictator Tests

Recall that a dictator function is a character x^i, for some $i \in [n]$, and we shall denote the set of dictators by $\mathcal{D}$. We shall find that the **dictator testing** problem, which asks us to determine whether an arbitrary Boolean-valued function is a dictator, is an incredibly versatile property test in the field of computing science.

A dictator test could surely start by testing for linearity. This would then reduce the problem to testing whether an arbitrary character x^S is a dictator. Classically, this was essentially the only method for constructing dictator tests. For instance, aside from the constant functions, the dictators are the only characters which satisfy

$$\mathrm{and}(x^S, y^S) = \prod_{i \in S} \mathrm{and}(x_i, y_i)$$

For an arbitrary S of size m, the probability of choosing x and y which fail this test is equal to the probability that uniformly chosen $T_0, T_1 \subset S$ are both odd, and $|T_0 \cap T_1|$ is odd, or that T_0, T_1 don't both contain an odd number of elements, and $|T_0 \cap T_1|$ is even. This is

$$(1/4^k) \left(\sum_{\substack{k \text{ odd} \\ k \leqslant m}} \sum_{\substack{l \text{ odd} \\ l \leqslant k}} \sum_{\substack{r \text{ even} \\ r \leqslant m-k}} \binom{m}{k} \binom{k}{2l+1} \binom{m-k}{r} \right)$$

TODO: BLAH BLAH FINISH UP LATER. And so we can test whether this equation holds to determine whether we have a dictator.

Using Arrow's theorem, and a bit of extra work, we can come up with a much more intelligent test. Given a Boolean-valued function f, take three random inputs X, Y, and Z uniformly from the 6 possible random triples (X, Y, Z) such that $\mathrm{NAE}(X, Y, Z)$ holds, and compute $\mathrm{NAE}(f(X), f(Y), f(Z))$. Accept f if the outputs are not all equal. We have shown that if f is accepted with probability $1 - \varepsilon$, then $\mathbf{W}^1(f) \geqslant 1 - (9/2)\varepsilon$, and the FKN theorem then implies that f is $O(\varepsilon)$ close to a dictator or its negation. However, this isn't exactly a dictator test, and requires a very difficult theorem to obtain a guarantee on closeness. Fortunately, if we do a NAE test and a BLR test sequentially, then we obtain a 6-query dictator test, and we don't even need to use the FKN test in our analysis to determine the local rate.

Theorem 4.1. *Suppose that $f : \{-1,1\}^n \to \{-1,1\}$ passes both the BLR test and the NAE test with probability $1-\varepsilon$. If $\varepsilon < 0.1$, then f is ε close to being a dictator. Alternatively, if $d(f,\mathcal{D}) > \varepsilon$, the for $\varepsilon < 0.1$, then f is rejected with probability greater than ε.*

Proof. We then know that f passes the BLR test with at least probability $1-\varepsilon$, hence there is S such that $d(f,x^S) \leqslant \varepsilon$, and f passes the NAE test with probability $1-\varepsilon$, hence $\mathbf{W}^1(f) \geqslant 1-(9/2)\varepsilon$. If S contains k elements, then $\mathbf{W}^k(f) \geqslant \widehat{f}(S)^2 \geqslant (1-2\varepsilon)^2 = 1-4\varepsilon+4\varepsilon^2$. If $k \neq 1$, then we must have

$$1-4\varepsilon+4\varepsilon^2 \leqslant (9/2)\varepsilon$$

which implies that $\varepsilon \geqslant 0.1$. If $k=1$, then $x^j \in \mathcal{D}$, hence $d(f,\mathcal{D}) \leqslant \varepsilon$. □

Thus in general, we find that if f passes the BLR test and NAE test with probability greater than $1-\varepsilon$, then f is 10ε close to a dictator, for if $\varepsilon < 0.1$, then f is ε close to a dictator, and if $\varepsilon \geqslant 0.1$, then f is $10\varepsilon \geqslant 1$ close to *any function*, hence a dictator of our choice. Thus the BLR and NAE test is a 0.1 rate 6-query local property tester. Note that we aren't that interested in finding local testers with low rejection rate, hence we are allowed to be sloppy with our analysis as above. Instead, we want to show that local testers exist for some property, and use as few queries to the function as possible.

There is a simple trick, given some property test for a class $\mathcal{C}$ with rate λ which employs numerous different subtests $T_1,\ldots,T_m$, is to pick *exactly one* of these tests at random, run the input on this test, and accept if the test passes. By a union bound, we then have

$$\mathbf{P}(\text{new test rejects } f) = (1/m)\sum \mathbf{P}(T_i \text{ rejects } f) \geqslant (1/m)\mathbf{P}(\text{old test rejects } f)$$

Suppose the old test has rejection rate λ. Then if $d(f,\mathcal{C}) > \varepsilon$, then the old test rejects f with probability greater than $\lambda\varepsilon$, and therefore the new test rejects f with probability $(\lambda/m)\varepsilon$. Thus the new test has rejection rate λ/m.

Example. *We have a* 0.1 *rate 6-query dictator test composed of two different tests, each applying 3-queries. Therefore we can use the reduction trick to obtain a* 0.05 *rate 3-query dictator test. Since the 6-query test is really rate* 1 *for* $\varepsilon < 0.1$, *the 3-query test is rate* 0.5 *for* $\varepsilon < 0.1$.

One nice property of dictatorship testing is that we can use a dictatorship test to obtain a 3-query property test over any subproperty $\mathcal{C} \subset \mathcal{D}$. Consider performing the following two tests on an input function f.

1. We perform a 3-query 0.05 rate dictatorship test on f.
2. Form the string y by $y_i = -1$ if $x^i \in \mathcal{C}$, and $y_i = 1$ otherwise. Then $y^j = -1$ if and only if $x^j \in \mathcal{C}$. We perform local decoding on f to calculate the value of f if it was linear.

We can combine the two tests to obtain a 3-query test over $\mathcal{C}$, but it is nicer to calculate the rejection rate assuming we perform both tests, and then double this rejection rate using the reduction trick. If the test passes with probability at least $1-\varepsilon$, then both the dictator test and the local decoding test pass with probability $1-\varepsilon$. If we let $\delta = 1$, for $\varepsilon < 0.1$, and $\delta = 20$, for $\varepsilon \geqslant 0.1$, we conclude that $d(f, x^j) \leqslant \delta\varepsilon$ for some dictator x^j. If $x^j \in \mathcal{C}$, we are done. Otherwise, if $x^j \notin \mathcal{C}$, then the local decoding calculates y^j from f accurately with probability at least $1 - 2\delta\varepsilon$, and we conclude that the test rejects f with probability at least $1 - 2\delta\varepsilon$. But then

$$1 > (1-\varepsilon) + (1 - 2\delta\varepsilon) = 2 - (1+2\delta)\varepsilon$$

hence $\varepsilon > 1/(1+2\delta)$. Thus if $\varepsilon \leqslant 0.1$, we conclude that f is ε close to a dictator, and so we have a 0.1 rate 6-query dictator test. Using the reduction trick, we obtain a 0.05 rate 3-query dictator test.

4.3 Probabilistically Checkable Proofs

We have shown that any subset of dictator functions has a 3 query proof. It turns out that testing over *any property* can be reduced to dictator testing, provided we have an appropriate 'proof'. To see this, we must slightly generalize our notion of property testing. Rather than just considering families of functions $f : \{-1,1\}^n \to \{-1,1\}$ to test over, we will be testing over binary strings of a fixed length – elements of $\{0,1\}^N$ for a fixed N. Rather than evaluated a function at positions, we get to randomly query certain bits of the string, and then choose to accept or reject the string. The rejection rates, query amount, and so on all generalize, because testing the value of a function at a certain point of the domain is equivalent to looking at a single bit of the binary string.

Example. *Consider testing whether a string is the all zero string. Given an input x, consider randomly selecting an index i, and then testing whether $x_i = 0$. If x is accepted with probability $1-\varepsilon$, then $1-\varepsilon$ percent of the bits in the string x are equal to 0, hence $d(x,0) = \varepsilon$, so this is a test with rate 1, and with a single query.*

Example. *What about testing if the values of a string are all the same. The natural choice is two queries – given x, we pick i and j and test whether $x_i = x_j$. If $\min(d(f,0), d(f,1)) = \varepsilon$, then*

$$\begin{aligned}\mathbf{P}(f \text{ is accepted}) &= \mathbf{P}(x_i = 0 | x_j = 0)\mathbf{P}(x_j = 0) + \mathbf{P}(x_i = 1 | x_j = 1)\mathbf{P}(x_j = 1) \\ &= d(f,0)^2 + d(f,1)^2 \\ &= 1 - 2d(f,0)[1 - d(f,0)]\end{aligned}$$

and so the test is rejected with probability $2\varepsilon(1-\varepsilon)$. If $\varepsilon < \alpha$, then the test is rejected with probability greater than $2(1-\alpha)\varepsilon$, so this is a local tester with rate $\max_{\alpha \in (0,1)} \min(\alpha, 2(1-\alpha)) = 2/3$.

Example. *Consider the property that a string has an odd number of 1s. Then the only local tester for this property must make all n queries.*

Now suppose that, in addition to $x \in \{0,1\}^n$, you are given some $\Pi \in \{0,1\}^m$ which is a 'proof' that x satisfies the required property. Then it might be possible to check both x and Π at the same time, *without ever having to trust the accuracy of* Π, and be able to obtain a higher rate algorithm that x is accepted, assuming that Π is the correct certificate. This is a **probabilistically checkable proof of proximity**. Specifically, an n-query, length m probabilisticlaly checkable proof of proximity system for a property $\mathcal{C}$ with rejection rate λ is an n-query test, which takes a bit string x, and a proof $\Pi \in \{0,1\}^m$, such that if $x \in \mathcal{C}$, there is a proof Π such that (x,Π) is always accepted, and if $d(x,\mathcal{C}) > \varepsilon$, then for *every* proof Π, (x,Π) is rejected with probability greater than $\lambda\varepsilon$. The converse is that if there is a proof Π such that (x,Π) is accepted with probability greater than $1-\varepsilon$, then $d(x,\mathcal{C}) \leqslant \varepsilon/\lambda$. We normally care about reducing n as much as possible, without regard to m or λ.

Example. *The property $\mathcal{O}$ that a string has an odd number of 1s cannot be checked with few queries without a proof, but we have an easy 3-query PCPP system with proof length $n-1$, and rejection rate 1. We expect the proof string to contain the sums $\Pi_j = \sum_{i \leqslant j+1} x_i$ in $\mathbf{F}_2$. We randomly check that one of the following equation holds*

- $\Pi_1 = x_1 + x_2$.
- $\Pi_j = \Pi_{j-1} + x_{j+1}$.
- $\Pi_{n-1} = 1$.

If $x \in \mathcal{O}$, and Π is a correct proof, then the test is accepted with probablity 1. If $x \notin \mathcal{O}$, then $d(x,\mathcal{O}) = 1/n$, and at least one of these equations fails, hence the probability that x is rejected is at least $1/n$. Thus this is a 3-query rate 1 PCPP tester.

Theorem 4.2. *Any property over length n bit strings has a 3-query PCPP tester with length-2^{2^n} proofs and rejection rate* 0.001.

Proof. Let $\mathcal{C}$ be the required property to test, and consider a bijection $\pi : [N] \to \{0,1\}^n$, where $N = 2^n$. Using π, each element of $\{0,1\}^n$ can be thought of as corresponding to a dictator on $[N]$. Given an input x, we will interpret a proof as a function $\Pi : \{0,1\}^N \to \{0,1\}$ as a dictator corresponding to x. The test then becomes a dictator test, which we can already solve with three queries. Thus we consider two separate tests.

- Testing if Π is a dictator in $\pi^{-1}(\mathcal{C})$.
- Assuming Π is a dictator corresponding to an element $y \in \{0,1\}^n$, testing whether $y = x$.

It is clear that if Π is a correct proof, and $x \in \mathcal{C}$, then the test will always be accepted. Otherwise, if (x,Π) is accepted with probability $1 - \varepsilon$, then the first test tells us that Π is 20ε close to being a dictator in three queries. It remains to find a clever way of determining which element of $\{0,1\}^n$ that the proof Π corresponds to, given that Π is close to some dictator $x^j \in \pi^{-1}(\mathcal{C})$. Local correcting tells us that any calculation we perform over Π will be correct to x_j with probability 40ε. Define $X_i^n = \pi(i)_n \in \{0,1\}^N$. If Π is close to the dictator x^j, then $(X^n)^j = \pi(j)_n$, and if $j = \pi^{-1}(x)$ then $(X^n)^j = x_n$ for all n. Thus our second test will pick n randomly, locally correct Π, and then test if $(X^n)^j = x_n$. If $j \neq \pi^{-1}(x)$, then x^j and $x^{\pi^{-1}(x)}$ agree on half their inputs, and since locally correcting Π to x^j is correct with probability greater than 40ε, our test will reject with probability greater than 80ε.

If Π is a correct proof, then $\Pi(x^i) = \chi_\alpha(x^i) \ldots$ finish later. □

The 2^{2^n} is a pretty bad estimate, but it is essentially required to be able to perform this test on all 2^{2^n} different properties on $\{0,1\}^n$. It should be reasonable to be able to identify better systems for 'explicit' properties. This can be qualified by the complexity of the circuit whic can compute the indicator function for the property.

A **PCPP reduction** is an algorithm which takes a property as input, and outputs a PCPP system testing that property. The notions of proof length, query size, etc all transfer to the PCPP reduction. Essentially, a PCPP reduction gives a constructive proof limiting the ability to test general properties. We have proved that there exists a 3-query 2^{2^n} proof length PCPP reduction with rejection rate 0.001. With a little work, we can improve this reduction to have proof length $2^{\text{poly}(C))}$, where the size of a property C is the size of the circuit representing it. There is a much deeper and more dramatic improvement to this property.

Theorem 4.3 (PCPP)**.** *There is a 3-query PCPP reduction with proof length polynomial in the size of the circuit required to represent the property.*

Though the 2^{2^n} PCPP reduction seems unnecessary after we have proved the PCPP theorem, the 2^{2^n} reduction is integral to the proof of the PCPP theorem, and is therefore important even if you want to understand the proof of the PCPP theorem.

4.4 Constraint Satisfaction and Property Testing

Recall that an m arity n contraint **satisfaction problem** over a set Ω is defined by a finite set of predicates $f_1, \ldots, f_n : \Omega^m \to \{0,1\}$. The goal is to find the elements of Ω^m which satisfy the most predicates. This is obviously NP complete, since the problem encompasses the SAT problem, so we instead try and find elements of Ω^m which satisfy the most predicates, or near to the most elements. We assume each f_i is an k-junta, for some positive integer k.

Example. *The 3-SAT problem on m variables is an m arity constraint satisfaction problem, where each constraint is a 3 junta.*

Example. *The MAX-CUT problem on a graph with n nodes and m edges is to find a partition of the graph so that the most possible edges cross the partitions. A partition corresponds to an element of assignment $\{0,1\}^n$, and this partition is evaluated against the not equal constraints for any edge (v,w).*

Example. *Given a system of linear equations in* $\mathbf{F}_2$*, each containing exactly 3 variabels, the MAX-E3-LIN problem asks to find an assignment satisfying as many of the constraints as possible.*

The k-query string testing and constraint satisfaction over k juntas are essentially the same problem. Given a constraint instance over Ω^m with constraints $f_1, \ldots, f_n$, and given a fixed solution $x \in \Omega^m$ (which we can view as a string input to test), consider the algorithm which randomly choosing $i \in [n]$ and accepts if $f_i(x) = 1$. The the probability the algorithm accepts x is $\text{val}(f, x)$. Conversely, if we have a string testing problem on Ω^n, with some randomized testing problem, then if we enumerate all random bits used in the algorithm, then each selection of random bits corresponds to a constraint where an element of Ω^n is tested, and the value of this constraint satisfaction problem gives the probability that the string testing algorithm is accepted.

4.5 Hastad's Hardness Theorem

The PCP theorem indicates that unless $P = NP$, it is impossible to approximate 3SAT to arbitrarily close constants. But we can find approximation algorithms for SAT for some constants. For instance, if each clause of MAX-E3SAT contains exactly 3 clauses (we call this subproblem MAX-E3SAT), then each clause is satisfied with probability $7/8$, hence in expectation a random assignment will satisfy $7/8$ of the clauses. There is a way to derandomize this algorithm to obtain a deterministic $7/8$ approximation algorithm. Though the approximation is trivial, Hastad's theorem says this is the best approximation possible.

Theorem 4.4 (Hastad)**.** *For any* $\delta > 0$*, if there is an algorithm that returns a 3SAT assignment satisfying at least* $7/8 + \delta$ *clauses for MAX-E3SAT for all instances of the problem where it is possible to satisfy all clauses, then* $P = NP$*.*

He also proved a similar optimal hardness of approximation for MAX-E3LIN, a result we will concentrate on for the rest of this section.

Theorem 4.5 (Hastad)**.** *If there is* $\delta > 0$*, and an approximation algorithm for MAX-E3LIN such that if* $1 - \delta$ *of the equations can be satisfied, then the algorithm returns an instance satisfying* $1/2 + \delta$ *of the equations, then* $\mathbf{P} = \mathbf{NP}$*.*

The idea of this theorem is that we can formalize a dictatorship test with a suitably high rejection rate by a reduction to an instance of MAX-E3LIN, and if we could approximate MAX-E3LIN to a factor greater than $1/2$, then we could solve the constraint satisfaction problem exactly, as we would therefore find $\mathbf{P} = \mathbf{NP}$.

We shall prove a slightly simpler form of this theorem. First, we introduce some formalization. Let $f_1, \ldots, f_n$ be predicates over $\{-1, 1\}$. If $0 < \alpha < \beta \leqslant 1$, let $\lambda : [0,1] \to [0,1]$ satisfy $\lambda(\varepsilon) \to 0$ as $\varepsilon \to 0$. Suppose that for each n there is a tester for functions $f : \{-1,1\}^n \to \{-1,1\}$ such that

- If f is a dictator, then the test accepts with probability at least β.
- If $\mathrm{Inf}_i^{1-\varepsilon}(f) \leqslant \varepsilon$, the test accepts with probability at most $\alpha + \lambda(\varepsilon)$.
- The tester can be viewed as an instance of MAX-CSP(f).

Then we call this family an **(α, β) Dictator vs. No Notables Test usings the predicates** f. It is essential that $\lambda = o(1)$, at a rate independent of n, but we care very little about the rate of convergence to zero. There is a Blackbox result which relates Dictator vs. No Notables to hardness of approximation results.

Theorem 4.6. *Fix a CSP over the domain $\{-1,1\}$ with predicates $f_1, \ldots, f_n$. Given an (α, β) dictator vs. no notables test using predicates f_i, then for all $\delta > 0$, either the universal game conjecture holds, or there is a $(\alpha + \delta, \beta - \delta)$ approximation for the CSP.*

Given a constraint satisfaction problem f, we define an (α, β) approximation algorithm to be the problem of coming up with an α approximation algorithm for the constraints f, subject to the constraint that we can assume that $\mathrm{val}(f) \geqslant \beta$. We rely on the following theorems, beyond our scope, to prove Hastad's theorem, which says that there is no $(1/2+\delta, 1-\delta)$ approximation algorithm for MAX-E3LIN unless the universal game conjecture holds. Because of the above theorem, it suffices to construct a $(1/2, 1)$ Dictator vs No Notables test using MAX-E3LIN constraints. This is exactly a test for some CSP accepts dictators almost surely, and if the stable influence $\mathrm{Inf}_i^{1-\varepsilon}(f) \leqslant \varepsilon$, then the test accepts with probability at most $1/2 + o(1)$, independent of n.

We cannot use the BLR test to find a $(1/2, 1-\delta)$ dictator vs no notables test using 3 variable $\mathbf{F}_2$ linear equations, because it doesn't accept functions with no notable coordinates with probability close to $1/2$. The two

problems are the constant 0 function and large parity functions. This is fixed by the odd BLR test. Given query access to $f : \{-1,1\}^n \to \{-1,1\}$, we pick $X, Y \in \{-1,1\}^n$ randomly, find $z \in \{-1,1\}$ randomly, and set $Z = XY(z,\ldots,z)$. We then accept the test if $f(X)f(Y)f(Z) = z$. It is easy to show that

Theorem 4.7. *The odd BLR test accepts f with probability*

$$\frac{1}{2} + \frac{1}{2}\sum_{|S| \text{ odd}} \widehat{f}(S)^3 \leqslant \frac{1}{2} + \frac{1}{2}\max_{|S| \text{ odd}} \widehat{f}(S)$$

Thus the odd BLR test rules out constant functions, but still accepts large parity functions. Hastad's idea was to add some noise to z on the left hand side of the equation, while still testing relative to z, so that the stability of high parity functions is knocked off. If f is a dictator, then f has high stability, and there is only a $\delta/2$ chance this will affect the test. On the other hand, if f is x^S, where S is large, there is a high chance this will disturb the chance of passing the test.

In detail, Hastad's δ test takes $X, Y \in \{-1,1\}^n$ and $b \in \{-1,1\}$ uniformly, sets $Z = b(XY)$, choses $Z' \sim N_{1-\delta}(Z)$, and accepts if $f(X)f(Y)f(Z') = (Z,\ldots,Z)$. We claim this is a $(1/2, 1-\delta/2)$ dictator vs. no notables test. We calculate

$$\begin{aligned}
\mathbf{P}(\text{Hastad accepts } f | b = 1) &= \mathbf{E}\left[\frac{1 + f(X)f(Y)f(Z')}{2}\right] \\
&= \frac{1 + \mathbf{E}[f(X)f(Y)T_{1-\delta}(f(XY))]}{2} \\
&= \frac{1 + \mathbf{E}[f(X)(f * T_{1-\delta}(f))(X)]}{2} \\
&= \frac{1}{2} + \frac{1}{2}\sum (1-\delta)^{|S|}\widehat{f}(S)^3
\end{aligned}$$

and

$$\begin{aligned}
\mathbf{P}(\text{Hastad accepts } f | b = -1) &= \mathbf{E}\left[\frac{1 - f(X)f(Y)f(Z')}{2}\right] \\
&= \frac{1}{2} - \frac{1}{2}\sum (-1)^{|S|}(1-\delta)^{|S|}\widehat{f}(S)^3
\end{aligned}$$

Hence the probability that the Hastad test accepts f is

$$\frac{1}{2} + \frac{1}{2} \sum_{|S| \text{ odd}} (1-\delta)^{|S|} \widehat{f}(S)^3$$

If $\mathrm{Inf}_i^{1-\varepsilon}(f) \leqslant \varepsilon$, we conclude that

$$\sum_{i \in S} (1-\varepsilon)^{|S|-1} \widehat{f}(S)^2 \leqslant \varepsilon$$

Hence

$$\begin{aligned}
\frac{1}{2} + \frac{1}{2} \sum_{|S| \text{ odd}} (1-\delta)^{|S|} \widehat{f}(S)^3 &\leqslant \frac{1}{2} + \frac{1}{2} \left(\sum_{|S| \text{ odd}} \widehat{f}(S)^2 \right) \left(\max_{|S| \text{ odd}} (1-\delta)^{|S|} \widehat{f}(S) \right) \\
&= \frac{1}{2} + \frac{1}{2} \sqrt{\max (1-\delta)^{2|S|} \widehat{f}(S)^2} \\
&= \frac{1}{2} + \frac{1}{2} \sqrt{\max (1-\delta)^{|S|-1} \widehat{f}(S)^2} \\
&= \frac{1}{2} + \frac{1}{2} \sqrt{\max_i \mathrm{Inf}_i^{1-\varepsilon}(f)} \\
&= \frac{1}{2} + \frac{1}{2} \sqrt{\varepsilon}
\end{aligned}$$

and $\sqrt{\varepsilon} \to 0$ as $\varepsilon \to 0$.

4.6 Max Cut Inapproximability

Recall that the MAX-CUT problem asks, given a graph G, to find the partition V, W of the vertices of the graph which maximizes the cardinality of edges crossing the cut, which is defined to be

$$\delta(V, W) = \{(v, w) \in E : v \in V, w \in W\}$$

As with many **NP**-complete optimization problems, MAX-CUT can be modelled as an instance of MAX-CSP over the alphabet $\{0, 1\}$, where the constraints are $v \neq w$, where $(v, w) \in E$. Semidefinite programming gives an α approximation algorithm for MAX-CUT, where

$$\alpha = \min_{0 < x < \pi} \frac{2x}{\pi(1 - \arccos x)}$$

We shall show that this is a tight bound for any approximation algorithm to MAX-CUT. As we have seen, the standard way to obtain an approximation bound for any k-CSP is to define a k query PCP verifier for some **NP** complete language L over an alphabet Σ^* obtained by considering a single predicate for MAX-CUT. If the PCP verifier accepts an element of L with probability greater than or equal to $1-\varepsilon$, and the PCP verifier rejects any other string with probability greater than $1-\delta$, then we obtain that MAX-CUT cannot have an α approximation algorithm for $\alpha > \delta/(1-\varepsilon)$ unless $\mathbf{P} = \mathbf{NP}$. However, our current knowledge of standard techniques for choosing the language L breaks down in the case where $k = 2$. One of the most important conjectures in complexity theory, second only to the $\mathbf{P} = \mathbf{NP}$ conjecture, is that there is a language L which is very applicable to 2-CSPs.

The problem UNIQUE-LABEL-COVER over an alphabet Σ is a 2-CSP defined with respect to a bipartite graph G with vertices $V \cup W$, such that for each edge (v,w) in the graph, we have a bijection $\pi_{(v,w)} : \Sigma \to \Sigma$, and the constraints are of the form $v = \pi_{(v,w)}(w)$, for each edge (v,w), where the variables are the vertices $V \cup W$ of the graph G. It is incredibly important result to most work in complexity theory that UNIQUE-LABEL-COVER is hard to approximate.

Theorem 4.8 (The Unique-Games Conjecture)**.** *For any $\eta, \gamma > 0$, there is an alphabet Σ such that it is* **NP** *hard to determine if an instance X of UNIQUE-LABEL-COVER over Σ has $val(X) \geqslant 1-\eta$, or $val(X) \leqslant \gamma$. That is, for any problem L in* **NP** *over strings in Π^*, there is a polynomial time computable function f taking strings in Π^* to instances of UNIQUE-LABEL-COVER over Σ, such that if $s \in L$ then $val(f(s)) \geqslant 1-\eta$, and if $s \notin L$ then $val(f(s)) \leqslant \gamma$.*

In this formulation, the unique game conjecture has a 2-query PCP verifier, such that each proof of an input x is encoded as an assignment to the 2-CSP $f(x)$, and we simply check if the assignment satisfies $v = \pi_{(v,w)}(w)$ for some random edge (v,w). Since f is polynomial time computable, we need only $O(\log n)$ random bits for this verifier.

The second 'conjecture' we will require was actually proved only in 2005, and is a statement about the properties of 'balanced' boolean functions.

Theorem 4.9 (Majority is Stablest)**.** *If $\rho \in [0,1)$, then for any $\varepsilon > 0$, there is $\delta > 0$ such that if $f : \{-1,1\}^n \to [-1,1]$ is an unbiased function, and $Inf_i(f) \leqslant$*

δ for all i, then

$$Stab_\rho(f) \leqslant 1 - \frac{2}{\pi}\arccos\rho + \varepsilon$$

hence among the balanced functions whose coordinates individually have small influence bounded by δ, we find that

$$Stab_\rho(f) \leqslant \lim_{\substack{n\to\infty \\ n\ odd}} Stab_\rho(Maj_n) + o_\delta(1)$$

hence the majority functions 'maximize' the stability of low influence functions, up to a $o_\varepsilon(1)$ factor.

Theorem 4.10. *The majority is stablest theorem continues to hold if we replace the assumption $Inf_i(f) \leqslant \delta$ with*

$$Inf_i^{\leqslant k}(f) = \sum_{\substack{i\in S \\ |S|\leqslant k}} \widehat{f}(S)^2 \leqslant \delta'$$

where k and δ' are universal functions of ε and ρ.

Proof. Fix $\rho < 1$ and $\varepsilon > 0$. If γ is chosen such that $\rho^k(1-(1-\gamma)^{2k}) < \varepsilon/4$ for all k, and δ is chosen such that if $\mathrm{Inf}_i(g) \leqslant \delta$ then $\mathrm{Stab}_\rho(g) \leqslant 1 - (2/\pi)\arccos\rho + \varepsilon/4$, then let $\delta' = \delta/2$ and choose k' such that $(1-\gamma)^{2k'} < \delta$. If f satisfies $\mathrm{Inf}_i^{\leqslant k'}(f) \leqslant \delta'$ and

Given $0 < \gamma < 1$, if we let $g = T_{1-\gamma}f$, and if we assume that for all i, $\mathrm{Inf}_i^{\leqslant k'}(f) \leqslant \delta'$, for some fixed δ', then we find

$$\mathrm{Inf}_i(g) \leqslant \mathrm{Inf}_i^{\leqslant k'}(f) + \sum_{\substack{i\in S \\ |S|>k'}} (1-\gamma)^{2|S|}\widehat{f}(S)^2 \leqslant \delta' + (1-\gamma)^{2k'}$$

If we choose γ small enough (depending only on δ, our choice of δ', and our choice of k'), then we find that $\mathrm{Inf}_i(g) \leqslant \delta$, and if we choose γ small enough such that $\rho^{|S|}[1-(1-\gamma)]^{2|S|} \leqslant \varepsilon$ for all $|S|$, we find

$$\mathrm{Stab}_\rho(f) = \mathrm{Stab}_\rho(g) + \sum \rho^{|S|}[1-(1-\gamma)]^{2|S|}\widehat{f}(S)^2 \leqslant \mathrm{Stab}_\rho(g) + \varepsilon$$

applying majority is stablest to $\mathrm{Stab}_\rho(g)$ gives the result for f. □

Theorem 4.11. *Majority is stablest implies that for any $\rho \in (-1, 0]$ and $\varepsilon > 0$, there is $\delta > 0$ such that for any boolean function $f : \{-1,1\}^n \to [-1,1]$, if $Inf_i(f) \leqslant \delta$ for all i, then*

$$Stab_\rho(f) \geqslant 1 - (2/\pi)\arccos\rho - \varepsilon$$

Proof. Given f, let g be the odd part of f, so $g(x) = [f(x) - f(-x)]/2$. Then $\mathbf{E}[g(x)] = 0$, $\mathrm{Inf}_i(g) \leqslant \mathrm{Inf}_i(f)$, and so we may apply majority is stablest for $-\rho$ to conclude

$$\begin{aligned}\mathrm{Stab}_\rho(f) &\geqslant \mathrm{Stab}_\rho(g) = -\mathrm{Stab}_{-\rho}(g) \\ &\geqslant -\left(1 - \frac{2}{\pi}\arccos(-\rho) + \varepsilon\right) \\ &= 1 - \frac{2}{\pi}\arccos(\rho) - \varepsilon\end{aligned}$$

hence the majority is stablest theorem holds for $\rho < 0$. □

Putting both these theorems together, we conclude that

Theorem 4.12. *For any $\rho \in (-1, 0]$ and $\varepsilon > 0$, there is $\delta > 0$ and k such that for any boolean function $f : \{-1,1\}^n \to [-1,1]$ with $Inf_i^{\leqslant k}(f) \leqslant \delta$ for all i, then*

$$Stab_\rho(f) \geqslant 1 - (2/\pi)\arccos\rho - \varepsilon$$

which is obtained by combining the two theorems above.

Now we construct a 2-query PCP verifier for UNIQUE-LABEL-COVER using MAX-CUT. By a refinement to the unique games conjecture, we can assume that in the instance of UNIQUE-LABEL-COVER is given over a bipartite graph G whose left vertices are regular, in the sense that they all have the same degree, so that choosing a random left vertex v, and then a random neighbour w corresponds to the uniform distribution on the set of edges in G. We assume the instance X of UNIQUE-LABEL-COVER has either $\mathrm{val}(X) \geqslant 1 - \eta$, or $\mathrm{val}(X) \leqslant \gamma$. The verifier expects the proof certificate Π to contain the long code $\Pi_w : \{-1,1\}^\Sigma \to \{-1,1\}$ for each right vertex w.

The 2-query PCP verifier using MAX-CUT with parameter $\rho \in (-1, 0)$

1: Pick some left vertex v at random, and then two neighbours w and w'.
2: Consider the permutations $\sigma = \sigma_{(v,w)}$, $\sigma' = \sigma_{(v,w')}$.
3: Pick random values $X, Y \in \{-1, 1\}^{\Sigma}$, with X uniform, and $Y \sim N_\rho(1)$.
4: Accept the input if $\Pi_w(X \circ \sigma) \neq \Pi_{w'}((X \circ \sigma')Y)$, else reject the input.

If Π_w and $\Pi_{w'}$ are actually the long codes of the labelling for w and w', where the labelling has value greater than or equal to $1-\eta$, then by a union bound we conclude that the constraints corresponding to (v, w) and (v, w') are both satisfied with probability greater than or equal to $1 - 2\eta$, and conditioning that this is true, the probability that the two are unequal is exactly the probability that $Y = -1$, which occur with probability $(1-\rho)/2$, hence the test accepts with probability $(1 - 2\eta)(1 - \rho)/2$. Conversely, the probability of accepting can be calculated to be exactly

$$\begin{aligned}
&\mathbf{P}(\Pi_w(X \circ \sigma) \neq \Pi_{w'}((X \circ \sigma')Y)) \\
&\quad = \mathbf{E}\left(\frac{1 - \Pi_w(X \circ \sigma)\Pi_{w'}((X \circ \sigma')Y)}{2}\right) \\
&\quad = \frac{1}{2} - \frac{1}{2}\mathbf{E}\left(\mathbf{E}[\Pi_w(X \circ \sigma)|X]\mathbf{E}[\Pi_{w'}((X \circ \sigma')Y)|X, v]\right) \\
&\quad = \frac{1}{2} - \frac{1}{2}\mathbf{E}\left[\mathrm{Stab}_\rho\left(\mathbf{E}[\Pi_w(X \circ \sigma)|X, v]\right)\right]
\end{aligned}$$

Denote $\mathbf{E}[\Pi_w(X \circ \sigma)|X, v] = g_v(X)$. It follows that if the probability of success is greater than or equal to $(\arccos\rho)/\pi + \varepsilon$, then by applying Markov's inequality, for at least $\varepsilon/2$ of the left vertices v,

$$\mathrm{Stab}_\rho(g_v) \leqslant 1 - (2/\pi)\arccos\rho - \varepsilon$$

and for each such vertex v (which we call call 'good' vertices), the majority is stablest theorem can be applied to determine that there is some index $s \in \Sigma$ such that $\delta \leqslant \mathrm{Inf}_s^{\geqslant k}(\Pi_v)$, and so

$$\delta \leqslant \sum_{\substack{s \in S \\ |S| \leqslant k}} \widehat{g_v}(S)^2 = \sum_{\substack{s \in S \\ |S| \leqslant k}} \mathbf{E}_w\left[\widehat{\Pi_w}(\sigma^{-1}(S))^2\right] = \mathbf{E}_w\left[\mathrm{Inf}_{\sigma^{-1}(s)}^{\leqslant k}(\Pi_w)\right]$$

For each $w \in W$, let the set C_w of candidate labels by all $s \in \Sigma$ such that $\mathrm{Inf}_s^{\leqslant k}(\Pi_w) \geqslant \delta/2$. Then $|C_w| \leqslant 2k/\delta$, and for each good vertex, v, at least

$\delta/2$ of the neighbours w of v have $\mathrm{Inf}^{\leqslant k}_{\sigma^{-1}(s)}(f) \geqslant \delta/2$, and so $\sigma^{-1}(s) \in C_w$. Now we consider a labelling of each vertex $w \in W$ by choosing an element of C_w, if possible, or any label if this set is empty. If follows that on the set of vertices adjacent to good vertices, at least $(\delta/2)(\delta/2k)$ are satisfied in expectation, and therefore there is a labelling which satisfies $(\varepsilon/2)(\delta/2)(\delta/2k)$ of the constraints. If $(\varepsilon/2)(\delta/2)(\delta/2k) > \gamma$, we conclude that the probability that the test accepts is less than $(\arccos\rho)/\pi + \varepsilon$.

The PCP verifier above shows that MAX-CUT is α hard to approximate, where

$$\alpha = \frac{2\arccos\rho + 2\varepsilon}{\pi(1-\rho)}$$

for any $\rho \in (-1, 0)$ and $\varepsilon > 0$. Maximizing over this set, letting $\varepsilon \to 0$ first and then optimizing, we conclude that we may let α be the hardness factor we first considered.

4.7 FOAIJDOIWJ

The standard way to obtain an approximation bound is to consider a PCP verifier for some **NP**-complete language L over some alphabet Σ^* using the predicates in MAX-CUT. Such a verifier leads to a polynomial time reducible function f taking strings in Σ^* to instances of MAX-CUT, in such a way that if $x \in L$, then $\mathrm{val}(f) \geqslant 1 - \varepsilon$, and if $x \notin L$ then $\mathrm{val}(f) \leqslant \delta$. If we had some α approximation algorithm for MAX-CUT, then unless $\mathbf{P} = \mathbf{NP}$, we would have to have $\alpha(1-\varepsilon) \leqslant \delta$, so $\alpha \leqslant \delta/(1-\varepsilon)$. However, it turns out that it is difficult

Recall that if J is an index set, then the long code encodes elements of J to Boolean valued function on $\{-1,1\}^J$, mapping $j \in J$ to the dictator function $x^j : \{-1,1\}^J \to \{-1,1\}$. Given $f : \{-1,1\}^J \to \{-1,1\}$, choose some $X \in \{-1,1\}^J$ at random, and let $Y \sim N_\rho(X)$. Our test will check whether $f(X) \neq f(Y)$. It is easy to arithmetize

$$\mathbf{P}(f(X) \neq f(Y)) = \mathbf{E}\left(\frac{1 - f(X)f(Y)}{2}\right) = \frac{1}{2} - \frac{1}{2}\mathrm{Stab}_\rho(f)$$

If f is a dictator, then $\mathbf{P}(f(X) \neq f(Y)) = (1-\rho)/2$. If f is not a dictator, then

$$\mathrm{Stab}_\rho(f) \geqslant 1 - \frac{2}{\pi}\arccos\rho$$

hence $\mathbf{P}(f(X) \neq f(Y)) \leqslant (1/\pi) \arccos \rho$.

Bibliography

[1] M. Bellare, D. Coppersmith, J. Håstad, M. Kiwi, M. Sudan. *Linearity Testing in Characteristic Two.*