Boolean Functions

Lecturers: Santosh Vempala and Anup Rao Scribe: Emma Cohen

March 8-10, 2016

Why boolean functions? Well, computers are boolean. But in terms of understanding complexity, we have some hope for boolean functions but it is still an interesting case.

1 Intro to boolean functions

First decision: for boolean values do we use $\{0, 1\}$ or $\{-1, 1\}$? There's no good consensus (mathematicians and computer scientsts lean in different directions), but we'll use the latter. (However, we will borrow addition mod 2 from the $\{0, 1\}$ setting, so addition on $\{-1, 1\}$ will be the same as multiplication).

Definition 1. A boolean function is a function f with domain $\{-1, 1\}^n$.

The codomain may be anything, but it will frequently be either $\{-1, 1\}$ (boolean valued) or $\mathbb{R}$ (real-valued).

Example 1.1 (Examples of boolean functions).

(a) $f(x) = (x_1 \vee x_2 \vee \bar{x}_3) \wedge (\dots)$

(b) $f(x) = \max\|x\|^2$ subject to $\sum_{i \neq j} x_i x_j = 1$

(c) We can think of a language (in the computing sense) as a collection of functions $f : \{-1, 1\}^n \to \{-1, 1\}$ (for various n).

We can think of a real-valued boolean function as a vector in $\mathbb{R}^{2^n}$ with coordinates indexed by corner of the hypercube. Then it is natural to define an inner product on this space:

Definition 2. The *inner product* of two boolean functions f, g w.r.t. distribution D on $\{-1, 1\}^n$ is

$$\langle f, g \rangle_D = \mathop{\mathsf{E}}_{x \sim D}(f(x)g(x))$$

If D is ommitted, we take it to be the uniform distribution over the hypercube.

Lemma 1.2. *If* $f, g, : \{-1, 1\}^n \to \{-1, 1\}$ *then* $\langle f, g \rangle = 1 - 2\,\mathsf{P}_D(f \neq g)$.

Proof. If $f(x) = g(x)$ then $f(x)g(x) = 1$, and otherwise it is -1. That is,

$$\langle f, g \rangle = \mathsf{P}(f = g) - \mathsf{P}(f \neq g) = 1 - \mathsf{P}(f \neq g).$$

□

We can also use this inner product in the usual way to define a norm on boolean functions: $\|f\|^2 = \langle f, f\rangle$. Note that if f is boolean-valued we always have $\|f\| = \mathsf{E}(f^2) = 1$ (since $f(x)^2 = 1$ for all x).

Thinking of boolean functions as vectors also lets us apply a change of basis. The standard basis is $e_1, \ldots, e_{2^n}$ with one basis vector for each corner of the boolean cube. This is of course an orthonormal basis.

Lemma 1.3. *For each set $S \subseteq \{-1,1\}^n$ define the parity function $\chi_S(x) := \prod_{i\in S} x_i$. The set of parity functions $\{\chi_S\}_{S\subseteq\{1,\ldots,n\}}$ is an orthonormal basis for the space of boolean functions.*

Proof. To show that they are a basis it suffices to see that the parity vectors are pairwise orthogonal (i.e., $\langle \chi_S, \chi_T\rangle = 0$ if $S \neq T$, so that they span a space of dimension 2^n (which must therefore be all of $\{-1,1\}^n$). Orthonormality only requires the additional property that $\|\chi_S\| = \langle \chi_S, \chi_S\rangle = 1$ for all S (which we have already observed, since the parity functions are boolean-valued). Expand

$$\begin{aligned}\langle \chi_S, \chi_T\rangle &= \mathsf{E}(\chi_S\chi_T) = \mathsf{E}(\chi_S(x)\chi_T(x)) \\ &= \mathsf{E}\Big(\prod_{i\in S} x_i \prod_{i\in T} x_i\Big) = \mathsf{E}\Big(\prod_{i\in S\setminus T} x_i \prod_{i\in T\setminus S} x_i \prod_{i\in S\cap T} (x_i)^2\Big) \\ &= \prod_{i\in S\setminus T} \mathsf{E}(x_i) \prod_{i\in T\setminus S} \mathsf{E}(x_i) \prod_{i\in S\cap T} (x_i)^2\end{aligned}$$

since the coordinates of a uniform random x are independent. Furthermore, since the coordinates of x have mean 0, if either of the first two sums is nonempty ($S \neq T$) then the whole product is 0, as desired. $\square$

Corollary 1.4. *Every boolean function f can be written uniquely as*

$$f(x) = \sum_{S\subseteq\{1,\ldots,n\}} \hat{f}(S)\chi_S(x).$$

The coefficients $\hat{f}$ are called the *discrete Fourier coefficients* and satisfy some nice properties.

Proposition 1.5.

1. $\hat{f}(S) = \langle f, \chi_S\rangle = \frac{1}{2^n}\sum f(x)\chi_S(x) = \mathsf{E}(f(x)\chi_S(x))$
2. *(Parseval's Theorem)* $\langle f, f\rangle = \sum_S \hat{f}(S)^2$
3. *(Plancherel's Theorem)* $\langle f, g\rangle = \sum_S \hat{f}(S)\hat{g}(S)$

Proof. These are all standard results about change of basis in a vector space applied to the boolean function setting (the first tells us how to perform the change of basis and the others say that inner products and lengths are preserved when changing from one orthonormal basis to another.) To see Plancherel's theorem directly (which of course implies Parseval's theorem), calculate

$$\begin{aligned}\langle f, g\rangle &= \Big\langle \sum_S \hat{f}(S)\chi_S, \sum_S \hat{g}(S)\chi_S\Big\rangle = \mathsf{E}\Big(\sum_{S,T} \hat{f}(S)\chi_S(x)\hat{g}(T)\chi_T(x)\Big) \\ &= \sum_{S,T} \hat{f}(S)\hat{g}(T)\,\mathsf{E}(\chi_S(x)\chi_T(x)) = \sum_S \hat{f}(S)\hat{g}(S)\end{aligned}$$

since $\mathsf{E}(\chi_S\chi_T) = 0$ whenever $S \neq T$ and 1 if $S = T$. $\square$

Definition 3. The *convolution* of f and g is

$$(f * g)(x) = \mathop{\mathsf{E}}_y(f(y)g(x-y))$$

(remember that addition and subtraction are both the same as multiplication in $\{-1, 1\}$.)

Lemma 1.6. *The Fourier coefficients of $f * g$ satisfy*

$$\widehat{(f * g)}(S) = \hat{f}(S)\hat{g}(S).$$

Proof.

$$\widehat{(f * g)}(S) = \langle f * g, \chi_S\rangle = \mathop{\mathsf{E}}_x\Big(\mathop{\mathsf{E}}_y(f(y)g(x-y))\chi_S(x)\Big) = \mathop{\mathsf{E}}_{x,y}(f(y)g(x-y)\chi_S(x))$$
$$= \mathop{\mathsf{E}}_{y,\, z=x-y}(f(y)g(z)\chi_S(z+y)) = \mathop{\mathsf{E}}_{y,z}(f(y)g(z)\chi_S(z)\chi_S(y)) = \hat{f}(S)\hat{g}(S)$$

since $z = x - y$ is uniform and independent of y and $\chi_S(x+y) = \chi_S(x)\chi_S(y)$. $\square$

2 Application to property testing

Problem 2.1. Test if a boolean function $f : \{-1,1\}^n \to \{-1,1\}$ is linear, i.e., $f(x+y) = f(x) + f(y)$ for all x, y.

Exercise 2.1. The only linear functions $\mathrm{GF}(2)^n \to \mathrm{GF}(2)$ are the parity functions.

Of course, in order to test whether a function (given by an oracle) is linear, we need to query every function value (exponentially many in the number of coordinates). What if we just want to be reasonably sure whether it's *close* to linear?

Algorithm 2.2 (Blum-Luby-Rubinfeld). Given an oracle for f, pick independent random x, y and test whether $f(x+y) = f(x) + f(y)$. If so, output YES (f is linear), otherwise output NO (f is not linear).

Of course, if f is linear, BLR always outputs YES. If f is far from being linear, then BLR outputs NO with high probability.

Proposition 2.3. *For the BLR algorithm above,* $\mathsf{P}(\text{YES}) \leq 1 - \mathrm{dist}(f, \mathcal{L})$*, where $\mathcal{L}$ is the set of all linear functions,* $\mathrm{dist}(f,g) = \mathsf{P}_x(f(x) \neq g(x))$*, and* $\mathrm{dist}(f, \mathcal{S}) = \min_{g\in\mathcal{S}} \mathrm{dist}(f, g)$ *is the distance from f to the closest function g in the set $\mathcal{S}$.*

That is, if f is ϵ-far from the nearest linear function then BLR outputs NO *with probability at least ϵ.*

Proof. We want to show that the probability that BLR outputs NO is at least the distance to the nearest linear function. By an earlier observation,

$$\langle f(x)f(y), f(x+y)\rangle = 1 - 2\mathop{\mathsf{P}}_{x,y}(f(x)f(y) \neq f(x+y)) = 1 - 2\,\mathsf{P}(\text{NO}).$$

On the other hand,

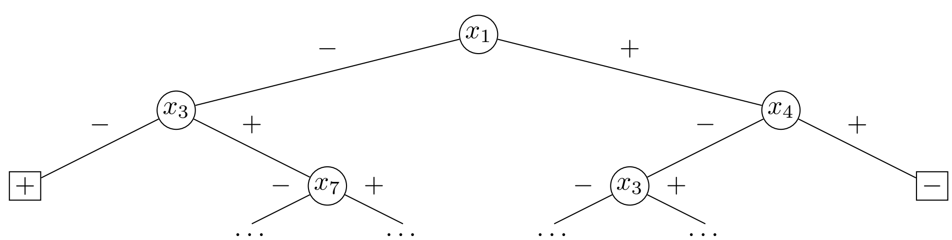
$$\begin{aligned}
\langle f(x)f(y), f(x+y)\rangle &= \mathop{\mathsf{E}}_{x,y}\Big(f(x)f(y)f(x+y)\Big) = \mathop{\mathsf{E}}_{x}\Big(f(x)\mathop{\mathsf{E}}_{y}(f(y)f(x-y))\Big) \\
&= \mathop{\mathsf{E}}_{x}(f(x)(f*f)(x)) = \langle f, f*f\rangle \\
&= \sum_S \hat{f}(S)\widehat{f*f}(S) = \sum_S \hat{f}(S)^3 \\
&\le \max_S \hat{f}(S)\sum_S \hat{f}(S)^2 \\
&= \max_S \hat{f}(S) = \max_S \langle f, \chi_S\rangle \\
&= \max_S(1-2\,\mathrm{dist}(f,\chi_S)) = 1 - 2\min_S \mathrm{dist}(f,\chi_S).
\end{aligned}$$

Putting these together gives $\min_S \mathrm{dist}(f,\chi_S) \le \mathsf{P}(\mathrm{NO})$, as desired. □

3 Application to learning

In the learning setting we talked about before, we were given random examples from some distribution D along with their labels and were supposed to come up with a way to classify new examples to minimize the probability of error (over new examples from D). This is hard to do outside of the case of linear classifiers, so let's give the learner more power: the ability to query the classification of particular examples it chooses. We also assume that the distribution D is uniform.

Recall that a decision tree determines a label for each boolean string by a path through the tree: each node is labeled by a variable and the child edge taken by the path is determined by the value of that variable; the leaves of the tree give the label for the corresponding path. For example:



Any boolean function has a decision tree, but most of them only have big decision trees (i.e., lots of leaves).

Suppose a decision tree has labeling function $f = \sum_S \hat{f}(S)\chi_S(x)$. If we approximate f by a labeling function g, then

$$\mathop{\mathsf{P}}_{x}(f(x) \neq g(x)) \le \frac{1}{4}\|f-g\|^2 = \frac{1}{4}\langle f-g, f-g\rangle = \frac{1}{4}\sum_S(\hat{f}(S)-\hat{g}(S))^2$$

That is, in order to approximate f it suffices to approximate $\hat{f}$ (in general, not just for decision trees). In the case of small decision trees, this approximation can be made to be sparse (even though in the standard basis f may be very far from sparse).

Let $\mathcal{S} = \{S \ : \ |\hat{f}(S)| \geq \epsilon/\|\hat{f}\|_1\}$ be the set of parity functions whose Fourier coefficients in f contribute at least an ϵ fraction of the total $\|\hat{f}\|_1 = \sum_S |\hat{f}(S)|$. Consider $g(x) = \sum_{S\in\mathcal{S}} \hat{f}(S)\chi_S(x)$. Then

$$1 - \langle f, g\rangle = 1 - \sum_{S\in\mathcal{S}} \hat{f}(S)^2 = \sum_{S\notin\mathcal{S}} \hat{f}(S)^2 \leq \frac{\epsilon}{\|\hat{f}\|_1} \sum_S |\hat{f}(S)| = \epsilon.$$

Note that $|\mathcal{S}| \leq \|\hat{f}\|_1/\epsilon$.

Lemma 3.1. *If f is described by a decision tree with m leaves, then $\|\hat{f}\|_1 \leq m$.*

Proof. Think of a leaf (or any path) in the tree as corresponding to the setting α of the variables occuring along the path, and let $\chi_\alpha(x) = 1$ if x agrees with α and 0 otherwise (so that χ_α is the indicator function for the subcube of inputs which agree with α). Write I_α for the set of indices of variables which occur along the path. If a leaf α has label ℓ_α, then we can write $f(x) = \sum_{\text{leaves } \alpha} \ell_\alpha \chi_\alpha(x)$. Note that

$$\hat{\chi}_\alpha(S) = \langle \chi_\alpha, \chi_S\rangle = \mathop{\mathsf{E}}_x(\chi_\alpha(x)\chi_S(x)) = \begin{cases} 0 & \text{if } S \not\subseteq I_\alpha \\ \chi_S(\alpha)\,\mathsf{E}(\chi_\alpha(x)) = \pm 2^{-|I_\alpha|} & \text{if } S \subseteq I_\alpha \end{cases}$$

So $\|\chi_\alpha\|_1 = \sum_{S\subseteq I_\alpha} 2^{-|I_\alpha|} = 1$. By the triangle inequality, $\|f\|_1 \leq \sum_{\text{leaves } \alpha} \|\chi_\alpha\|_1 = m$. □

That is, to approximate the labeling function f of a decision tree with m leaves to within error ϵ, we just need to find the largest m/ϵ Fourier coefficients of f. Since we have an oracle for f, we can query random values of f and use the results to approximate $\hat{f}(S) = \mathsf{E}(f\chi_S)$.

Of course, there are exponentially many coefficients to look at, so we will need to do better. Instead of just estimating a particular Fourier coefficient, we can estimate the sum of squared Fourier coefficients

$$w_\alpha = \sum_{S\in\mathcal{S}_\alpha} \hat{f}(S)^2$$

for any prefix $\alpha \in \{0,1\}^k$, where $\mathcal{S}_\alpha = \{S \subseteq [n] : \ 1_{S\cap[k]} = \alpha\}$ is the collection of subsets which agree with α on the first k indices. Then we can perform a binary search for the sets with large Fourier coefficients: If $w_\alpha < (\epsilon/m)^2$ then no $S \in \mathcal{S}_\alpha$ is important; otherwise we split $\mathcal{S}_\alpha = \mathcal{S}_{\alpha 0} \cup SS_{\alpha 1}$ and recurse on those sets (or if $k = n$ we have found a large Fourier coefficient). Since there are no more than m large Fourier coefficients, this only requires us to estimate a polynomial number of weights w_α. Thus it only remains to see how to estimate w_α.

Lemma 3.2. *For any prefix α of length k,*

$$\sum_{S\subseteq S_\alpha} \hat{f}(S)^2 = \mathop{\mathsf{E}}_{x,y,z}(f(yx)f(zx)\chi_\alpha(y)\chi_\alpha(z))$$

where $x \sim \{\pm 1\}^{n-k}$ and $y, z \sim \{\pm 1\}^k$ (and zx represents the concatenation of z and x).

Proof. Note that

$$\mathop{\mathsf{E}}_{x,y,z}(\chi_T(yx)\chi_T(zx)\chi_\alpha(y)\chi_\alpha(z)) = \begin{cases} 1 & T \in \mathcal{S}_\alpha \\ 0 & \text{otherwise.} \end{cases}$$

For general f,

$$\mathop{\mathsf{E}}_{x,y,z}(f(yx)f(zx)\chi_\alpha(y)\chi_\alpha(z)) = \mathop{\mathsf{E}}_{x,y,z}\Big(\sum_{T,T'}\hat{f}(T)\hat{f}(T')\chi_T(yx)\chi_{T'}(zx)\chi_\alpha(y)\chi_\alpha(z)\Big)$$
$$= \sum_{T,T'}\hat{f}(T)\hat{f}(T')\mathop{\mathsf{E}}_{x,y,z}(\chi_T(yx)\chi_{T'}(zx)\chi_\alpha(y)\chi_\alpha(z))$$
$$= \sum_{T}\hat{f}(T)^2\mathop{\mathsf{E}}_{x,y,z}(\chi_T(yx)\chi_T(zx)\chi_\alpha(y)\chi_\alpha(z))$$
$$= \sum_{T\in\mathcal{S}_\alpha}\hat{f}(T)^2$$

since $\mathsf{E}(\chi_T(yx)\chi_{T'}(zx)\chi_\alpha(y)\chi_\alpha(z)) = 0$ whenever $T \neq T'$. □

So to query $\sum_{S\subseteq S_\alpha}\hat{f}(S)^2$, we can just estimate the expectation by sampling

$$f(yx)f(zx)\chi_\alpha(y)\chi_\alpha(z)$$

for random x, y, z (since this is bounded between ± 1 the average converges quickly to the expectation).

4 Boolean functions as voting schemes

Note that $\mathsf{E}(f) = \hat{f}(\emptyset)$ and $\mathsf{Var}(f) = \sum_{S\neq\emptyset}\hat{f}(S)^2$. In particular, if f is boolean-valued and $\mathsf{E}(f) = 0$ then $\mathsf{Var}(f) = 1$.

Suppose there are two candidates in an election with n voters. We can think of the votes as $x \in \{-1,1\}^n$ and a boolean function f as a scheme for aggregating votes. For example, one might take f to be the majority function $\mathsf{Maj}(x) = \text{sign}(x_1 + x_2 + \cdots + x_n)$ or a dictatorship $\mathsf{Dict}_i(x) = x_i$.

Some properties we might want in an election system are

1. Monotonicity. If $x_i \leq y_i$ for all i then $f(x) \leq f(y)$ (so changing your vote from candidate $-$ to candidate $+$ cannot hurt candidate $+$)
2. Oddness. $f(-x) = -f(x)$
3. Symmetry. For all x and permutations π, $f(x) = f(x^\pi)$, i.e., permuting the coordinates of x by $\pi \in S_n$ does not change the result.

Majority satisfies all of these.

Definition 4. For a bit string $x \in \{-1,1\}^n$ and $\rho \in [0,1]$, define the distribution $y \sim N_\rho(x)$ by

$$y_i = \begin{cases} x_i & \text{w/ prob. } \rho \\ \pm 1 & \text{w/ prob. } 1-\rho \end{cases} = \begin{cases} x_i & \text{w/ prob. } \frac{1+\rho}{2} \\ -x_i & \text{w/ prob. } \frac{1-\rho}{2} \end{cases}.$$

In fact, this second formula works for all $\rho \in [-1,1]$, so we'll use that one for the remainder of the discussion. Say (x,y) are *ρ-correlated* if x is uniform and $y \sim N_\rho(x)$. (Note that in the second definition we can also take $\rho \in [-1,0)$ to get a negative correlation between x and y.)

Definition 5. The *noise operator* is $T_\rho f(x) = \mathsf{E}_{y \sim N_\rho(x)} f(y)$. The *noise stability* of f at rate ρ is

$$\mathrm{Stab}_\rho(f) = \langle f, T_\rho f \rangle = \mathop{\mathsf{E}}_{\substack{x,y \\ \rho\text{-correlated}}} (f(x)f(y)).$$

For example,

- The stability of dictatorship is $\mathrm{Stab}_\rho(\mathsf{Dict}_i) = \langle x_1, T_\rho(x_1) \rangle = \rho$ (since with probability ρ, $T_\rho x_1 = x_1$, and otherwise it is uniform random).
- $T_\rho(\chi_S) = \rho^{|S|}\chi_S$, so the stability of a parity function is $\mathrm{Stab}_\rho(\chi_S) = \rho^{|S|}$, since if any index of S is randomized the expectation is 0.

Using this we can calculate

Lemma 4.1.

$$T_\rho f = \sum_S \hat{f}(S)\rho^{|S|}\chi_S(x)$$

and

$$\mathrm{Stab}_\rho(f) = \sum_S \rho^{|S|}\hat{f}(S)^2 = \sum_{k=0}^{n} \rho^k w_k,$$

where $w_k \triangleq \sum_{|S|=k} \hat{f}(S)^2$.

That is, T_ρ dampens the Fourier coefficients for large sets.

4.1 Condorcet voting

For an m-candidate election, suppose every voter has a preference between every pair of candidates. Say that a preference is *rational* if it is transitive, i.e., if A is preferred to B and B is preferred to C then A is also preferred to C (so a rational voter's preferences are consistent with a linear order on the candidates). If we have many voters, we can also find the majority preference for every pair of candidates. Condorcet noted that even if the individual voters are rational, the aggregate majority preference may not be. Unfortunately this is not just a problem with majority aggregation, as we will see.

We assume that a vote aggregation scheme has the property that the aggregate preference between candidates A and B only depends on the voters' preferences between A and B (and not the other candidates) and that it is symmetric (all candidates are compared in the same way), so there is some boolean function f such that if the voters' preferences between A and B are given by $x \in \{\pm 1\}^n$ then the aggregate preference between A and B is $f(x)$ (for each pair of candidates).

Theorem 4.2 (Arrow's theorem). *Suppose f is a rational vote aggregation scheme (i.e., if the individual voters are rational then so is the aggregate preference), f is unanimous (i.e., $f(-1, \ldots, -1) = -1$ and $f(1, \ldots, 1) = 1$), and f is unbiased ($\mathsf{E}(f) = 0$). Then $f = \mathsf{Dict}_i$ for some i.*

The proof of Arrow's theorem is left as an exercise. We will actually give a stronger result. It is sufficient to consider the case of three candidates, where each voter's preferences can be expressed in terms of the three pairwise preferences (A vs. B, B vs. C, and C vs. A). If $(x_i, y_i, z_i) \in \{\pm 1\}^3$ denote the ith voter's preferences, then the voter is rational iff $\mathsf{NAE}(x_i, y_i, z_i) = 1$ (where $\mathsf{NAE}(x, y, z) = 1$ if x, y, z are not all equal and 0 otherwise). The result then follows from the lemma (whose proof is left as an exercise)

Lemma 4.3 (Kalai [3]). *For any boolean function f,*

$$\mathsf{E}(\mathsf{NAE}(f(x), f(y), f(z))) = \frac{3}{4} - \frac{3}{4}\,\mathrm{Stab}_{-\frac{1}{3}}(f)$$

where the expectation is over x, y, z which are rational, i.e. $\mathsf{NAE}(x_i, y_i, z_i) = 1$ for all i.

Once we have this, for a rational aggregation scheme we can write

$$\begin{aligned} 1 &= \frac{3}{4} - \frac{3}{4}\,\mathrm{Stab}_{-\frac{1}{3}}(f) = \frac{3}{4} - \frac{3}{4}\sum_{k=0}^{n}(-\tfrac{1}{3})^k w_k \\ &\le \frac{3}{4} + \frac{3}{4}\left(\frac{1}{3}w_1 + \frac{1}{9}(1 - w_1)\right) = \frac{7}{9} + \frac{2}{9}w_1 \end{aligned}$$

since $-(-3)^{-k} \le \frac{1}{9}$ when $k = 0$ or $k > 1$. Rearranging this gives

$$1 \le w_1 = \sum_{i=1}^{n} \hat{f}(i)^2.$$

The only boolean (integer) functions satisfying this have $\hat{f}(i) = \pm 1$ for some i. Unanimity says that it must be $\hat{f}(i) = 1$ and so $f = \mathsf{Dict}_i$. More generally,

Theorem 4.4 (Friedgut-Kalai-Naor [1]). *Suppose $f : \{-1,1\}^n \to \{-1,1\}$ has $w_1(f) \ge 1 - \delta$ then f is $O(\delta)$-close to a dictatorship.*

We now quantify the influence that each voter has over the outcome of the election. Intuitively, in a fair election no one voter (or small group of voters) should have too much influence.

Definition 6. The *influence* of the ith coordinate in $f : \{-1,1\}^n \to \{-1,1\}$ is

$$\mathrm{Inf}_i(f) = \mathop{\mathsf{P}}_{x}(f(x) \ne f(x^{\oplus i}))$$

where $x^{\oplus i}$ is x with the ith bit flipped. This is just the probability (over a uniform choice of other votes) that the ith voter can change the outcome of the election. We also define

$$\mathrm{Inf}(f) = \sum_i \mathrm{Inf}_i(f).$$

Definition 7. The *derivative* of f in direction i is $D_i f : \{-1,1\}^n \to \{-1,0,1\}$ by

$$D_i f(x) = \frac{f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n) - f(x_1, \ldots, x_{i-1}, -1, x_{i+1}, \ldots, x_n)}{2}.$$

We can also write the influence in terms of the derivative:

$$\mathrm{Inf}_i(f) = \mathop{\mathsf{E}}_x((D_i f(x))^2) = \langle D_i f, D_i f\rangle.$$

For example,

- $\mathrm{Inf}_i(\mathsf{Dict}_j) = 1$ if $i = j$ and 0 otherwise.
- $\mathrm{Inf}_i(\mathsf{Maj}_n) = \frac{1}{2^n}\binom{n-1}{(n-1)/2} = \Theta(1/\sqrt{n})$.

Remember that we can write $\mathsf{Var}(f) = \sum_{k=1}^n w_k$. We can calculate

Lemma 4.5.

1. *$D_i\chi_S = \chi_{S\setminus i}$ if $i \in S$, and 0 otherwise,*
2. *$D_i f = \sum_S \hat{f}(S) D_i \chi_S = \sum_{S\ni i} \hat{f}(S)\chi_{S\setminus i}$,*
3. *$\mathrm{Inf}_i(f) = \sum_{S\ni i} \hat{f}(S)^2$, and*
4. *$\mathrm{Inf}(f) = \sum_i \mathrm{Inf}_i(f) = \sum_S \sum_{i\in S} \hat{f}(S)^2 = \sum_{k=1}^n k\, w_k \geq \mathsf{Var}(f)$.*

Note that if $\mathsf{E}(f) = 0$ then $\mathsf{Var}(f) = 1$ and so

Lemma 4.6 (Poincaré Inequality). *There is some index i with $\mathrm{Inf}_i(f) \geq 1/n$.*

Write $\mathrm{MInf}(f) = \max_i \mathrm{Inf}_i(f)$ for the maximum influence. We have just seen that $\mathrm{MInf}(f) \geq 1/n$, but in fact this is not quite tight.

Theorem 4.7 (Kahn-Kalai-Linial [2]). *If f is an unbiased voting scheme, then*

$$\mathrm{MInf}(f) \geq \Omega\left(\frac{\log n}{n}\right).$$

If f is biased, one can still show that $\mathrm{MInf}(f) \geq \mathsf{Var}(f)\Omega(\log n/n)$, but we will only prove the unbiased case.

Proof of KKL. It suffices to prove $\mathrm{MInf}(f) \geq \frac{9}{\mathrm{Inf}(f)^2} \cdot 9^{-\mathrm{Inf}(f)}$, because if $\mathrm{Inf}(f) \geq c \log n$ then the result follows immediately, and otherwise $\mathrm{MInf}(f) \geq \frac{9n^{-c\log 9}}{(c\log n)^2}$. By choosing $c < 1/\log 9$, we ensure that each of these bounds satisfies the bound in the theorem.

To prove this, we use the following lemma, whose proof is left as an exercise (or see [4]):

Lemma 4.8 (Bonami). $\|T_{\sqrt{\frac{1}{3}}}\, g\|_2^2 \leq \|g\|_{4/3}^2$.

We will prove that $\mathrm{Inf}(f)\sqrt{\mathrm{MInf}(f)} \geq 3 \cdot 3^{-\mathrm{Inf}(f)}$ by sandwiching $\sum_{i=1}^n \mathrm{Stab}_{\frac{1}{3}}(D_i f)$ between the two sides. On one side we have

$$\sum_{i=1}^n \mathrm{Stab}_{\frac{1}{3}}(D_i f) = \sum_{i=1}^n \sum_{S\ni i} (\tfrac{1}{3})^{|S|-1}\hat{f}(S)^2 = 3\sum_{k=1}^n k(\tfrac{1}{3})^k w_k$$
$$\geq 3\sum_{k=1}^n (\tfrac{1}{3})^k w_k = 3\,\mathrm{Stab}_{\frac{1}{3}} f = 3\sum_{k=0}^n (\tfrac{1}{3})^k w_k.$$

Since 3^{-x} is convex and the coefficients w_k are a convex combination,

$$\sum_{k=0}^{n} 3^{-k} w_k \;\geq\; 3^{-\sum_k k w_k} = 3^{-\operatorname{Inf}(f)}.$$

For the other side, note that $\operatorname{Stab}_\rho(g) = \langle g, T_\rho\, g\rangle = \langle T_{\sqrt{\rho}}\, g, T_{\sqrt{\rho}}\, g\rangle$, so by Bonami

$$\begin{aligned}\operatorname{Stab}_{\frac{1}{3}}(D_i f) &= \|T_{\sqrt{\frac{1}{3}}}\, D_i f\|_2^2 \\ &\leq \|D_i f\|_{4/3}^2 = \left(\mathsf{E}\big(|D_i f|^{4/3}\big)\right)^{3/2} = \left(\mathsf{E}\big((D_i f)^2\big)\right)^{3/2} = \operatorname{Inf}_i(f)^{3/2}\end{aligned}$$

and thus

$$\sum_i \operatorname{Stab}_{\frac{1}{3}}(D_i f) \leq \sum_i \operatorname{Inf}_i(f)^{3/2} \leq \left(\max_i \sqrt{\operatorname{Inf}_i(f)}\right) \sum_i \operatorname{Inf}_i(f) = \sqrt{\operatorname{MInf}(f)}\, \operatorname{Inf}(f).$$

□

References

[1] Ehud Friedgut, Gil Kalai, and Assaf Naor. Boolean functions whose fourier transform is concentrated on the first two levels. *Advances in Applied Mathematics*, 29(3):427 – 437, 2002.

[2] J. Kahn, G. Kalai, and N. Linial. The influence of variables on boolean functions. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science*, SFCS '88, pages 68–80, Washington, DC, USA, 1988. IEEE Computer Society.

[3] Gil Kalai. A fourier-theoretic perspective on the condorcet paradox and arrow's theorem. *Advances in Applied Mathematics*, 29(3):412 – 426, 2002.

[4] Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.