

## Lecture 10 : March 29, 2007

*Lecturer: Rocco Servedio**Scribe: Ben London***Previous Lecture**

In the previous lecture, we introduced the concept of noise sensitivity, and used it to prove that intersections of halfspaces are learnable in polynomial time, using the Low-Degree Algorithm. One should recall the following definition of noise sensitivity.

**Definition 1** For any boolean function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ , the noise sensitivity,  $ns_\epsilon(f)$ , is defined as the probability that the output of the original input will disagree that of the noisy input,  $N_\epsilon(x)$ .

$$ns_\epsilon(f) = \Pr[f(x) \neq f(N_\epsilon(x))]$$

In this lecture, we will see two more applications of the Low-Degree Algorithm. First, we will continue our discussion of halfspaces, focusing on intersections of arbitrary functions of halfspaces. Following this, we will introduce the concept of *influence*, as it relates to learning monotone functions, and use it to arrive at a  $2^{\tilde{O}(\sqrt{n})/\epsilon}$ -time algorithm for learning monotone functions.

## 1 Learning Intersections of Arbitrary Functions of Halfspaces

Our second application of the Low-Degree Algorithm is for learning arbitrary functions of halfspaces. Recall the following lemmas from the previous lecture.

**Lemma 1** (Peres) Any halfspace  $h : \{-1, 1\}^n \rightarrow \{-1, 1\}$  has noise sensitivity,

$$ns_\epsilon(h) \leq 8.8\sqrt{\epsilon}$$

.

**Lemma 2** For any boolean function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ , if you have that  $ns_\epsilon(h) \leq \beta(f)$ , then you have Fourier concentration,

$$\sum_{|S| \geq 1/\beta^{-1}(\frac{\epsilon}{2.32})} \hat{f}(S)^2 \leq \epsilon$$

For a halfspace,  $f$ , we have that  $\beta(\epsilon) = 8.8\sqrt{\epsilon}$ . Therefore,  $\beta^{-1}(x) = (\frac{x}{8.8})^2$  and,

$$\sum_{|S| \geq (\frac{8.8(2.32)}{\epsilon})^2} \hat{f}(S)^2 \leq \epsilon$$

We now consider a set of halfspaces,  $h_1, \dots, h_k$ . Let  $g(x_1, \dots, x_k)$  be any function of  $k$  boolean variables, and let  $f(x) = g(h_1(x), \dots, h_k(x))$ .

**Proposition 1** *The noise sensitivity of  $f$  is bounded by,*

$$ns_\epsilon(f) \leq 8.8k\sqrt{\epsilon}$$

**Proof:** Recall the definition of noise sensitivity.

$$ns_\epsilon(f) = \Pr[f(x) \neq f(N_\epsilon(x))]$$

Expanding  $f$ , we have,

$$\Pr[f(x) \neq f(N_\epsilon(x))] = \Pr[g(h_1(x), \dots, h_k(x)) \neq g(h_1(N_\epsilon(x)), \dots, h_k(N_\epsilon(x)))]$$

Therefore, taking the union bound over all  $k$  halfspaces, we arrive at,

$$\begin{aligned} \Pr[f(x) \neq f(N_\epsilon(x))] &\leq \sum_{i=1}^k \Pr[h_i(x) \neq h_i(N_\epsilon(x))] \\ &\leq 8.8k\sqrt{\epsilon} \end{aligned}$$

■

Now we know that, for any function  $f$  of  $k$  halfspaces, we have  $ns_\epsilon(f) \leq \beta(\epsilon) = 8.8k\sqrt{\epsilon}$ . From this, we can say that  $\beta^{-1}(x) = \left(\frac{x}{8.8k}\right)^2$ . Therefore, we can bound the Fourier concentration of arbitrary functions of  $k$  halfspaces by,

$$\sum_{|S| \geq \left(\frac{8.8(2.32)k}{\epsilon}\right)^2} \hat{f}(S)^2 \leq \epsilon$$

Thus, we know that the Low-Degree Algorithm can learn any arbitrary function of  $k$  halfspaces in  $n^{O(k/\epsilon^2)}$ -time.

**Open Question 1** *Given a conjunction of  $k$  halfspaces,  $f(x) = h_1(x) \wedge \dots \wedge h_k(x)$ , is the noise sensitivity  $ns_\epsilon(f) = O(\log k)\sqrt{\epsilon}$ ?*

## 2 Learning Monotone Functions

Our third, and final, application of the Low-Degree algorithm is for learning monotone functions. But before launching into this topic, we must define the concept of *influence*.

### 2.1 Influence

**Definition 2** *Given a boolean function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ , the influence of variable  $i$  on  $f$  is defined as the probability that the function of the original input will disagree with that of an input with the  $i^{\text{th}}$  bit flipped. We denote the influence of a single bit by,*

$$Inf_i(f) = \Pr[f(x) \neq f(x^{\oplus i})]$$

where  $x^{\oplus i}$  is the input string  $x$  over  $\{-1, 1\}^n$  with the  $i^{\text{th}}$  bit flipped. We define the overall influence of  $f$  as,

$$Inf(f) = \sum_{i=1}^n Inf_i(f)$$

### Examples of Influence

1. The constant function,  $f(x) = 1$ : Since no variable has any effect on the output of  $f$ ,

$$\begin{aligned} \text{Inf}_i(f) &= 0, \forall i \\ \implies \text{Inf}(f) &= 0 \end{aligned}$$

2. The parity function,  $f(x) = \text{PAR}(x)$ : The parity function exhibits the maximum possible influence, as flipping any variable will undoubtedly flip the output. Therefore,

$$\begin{aligned} \text{Inf}_i(f) &= 1, \forall i \\ \implies \text{Inf}(f) &= n \end{aligned}$$

3. Conjunctions,  $f(x) = x_1 \wedge \dots \wedge x_n$ : For a single bit to change the value of a conjunction, we must have the condition in which all of the other  $n-1$  variables equal 1. Otherwise, if any of the others equalled  $-1$ , flipping the  $i^{\text{th}}$  bit would not matter, because the output would already be  $-1$ . Therefore, since the occurrence of this condition happens with probability  $\frac{1}{2^{n-1}}$ , we have,

$$\begin{aligned} \text{Inf}_i(f) &= \frac{1}{2^{n-1}}, \forall i \\ \implies \text{Inf}(f) &= \frac{n}{2^{n-1}} \end{aligned}$$

4. Decision List-type functions: The influence of a decision list-type function variable is similar to that of a conjunction, with the exception that the order of the variables plays a role in determining influence. A variable  $x_i$  is only evaluated when the preceding  $i-1$  variables are not satisfied. Therefore, since the occurrence of this condition happens with probability  $\frac{1}{2^{i-1}}$ , we have,

$$\begin{aligned} \text{Inf}_i(f) &\leq \frac{1}{2^{i-1}}, \forall i \\ \implies \text{Inf}(f) &\leq \sum_{i=1}^n \frac{1}{2^{i-1}} \end{aligned}$$

5. The majority function,  $f(x) = \text{MAJ}(x) = \text{sign}(x_1 + \dots + x_n)$ : For the majority function, the only condition in which a single variable would have influence over the output would be in which the rest of the variables are split in half between  $-1$  and  $1$ . We can calculate this probability using Bernoulli, which gives us,

$$\begin{aligned} \text{Inf}_i(f) &= \binom{n-1}{\frac{n-1}{2}} \frac{1}{2^{n-1}} = \Theta\left(\frac{1}{\sqrt{n}}\right), \forall i \\ \implies \text{Inf}(f) &= n\Theta\left(\frac{1}{\sqrt{n}}\right) = \Theta(\sqrt{n}) \end{aligned}$$

Note that the majority function has the highest influence of any monotonic function. (We'll prove this later.)

So, what about Fourier?

**Proposition 2** *Given a boolean function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ , the influence of any variable  $i$  is equal to the sum of Fourier coefficients, for sets which contain  $i$ .*

$$\text{Inf}_i(f) = \sum_{S:i \in S} \hat{f}(S)^2$$

**Proof:** We begin by defining a function  $f^{(i)}(z)$ , as the difference of a boolean function of a normal input and that of a bit-flipped input.

$$f^{(i)}(z) = f(z) - f(z^{\oplus i})$$

Note that  $f^{(i)}(z)$  takes values over  $\{-2, 0, 2\}$ . If  $f(z) = f(z^{\oplus i})$  then  $f^{(i)}(z) = 0$ . Otherwise, it will output a value in  $\{-2, 2\}$ . Thus, the influence is the probability of a  $\pm 2$  output. Formally, we have,

$$\begin{aligned} \text{Inf}_i(f^{(i)}) &= \Pr[f(x) \neq f(x^{\oplus i})] \\ &= \Pr[f^{(i)}(x) \neq 0] \\ &= \frac{1}{2^n} (\text{number of } x\text{'s s.t. } f^{(i)}(x) \neq 0) \\ &= \frac{1}{2^n} \sum_x \frac{f^{(i)}(x)^2}{4} \\ &= \frac{1}{4} \left( \frac{1}{2^n} \sum_x f^{(i)}(x)^2 \right) \\ &= \frac{1}{4} \mathbb{E}[f^{(i)}(x)^2] \end{aligned}$$

We can now utilize Parseval's theorem to arrive at,

$$\frac{1}{4} \mathbb{E}[f^{(i)}(x)^2] = \frac{1}{4} \sum_S \hat{f}^{(i)}(S)^2$$

By definition,

$$\hat{f}^{(i)}(S) = \mathbb{E}[f^{(i)}(x) \chi_S(x)]$$

Therefore, we have,

$$\begin{aligned} \frac{1}{2^n} \sum_x f^{(i)}(x) \chi_S(x) &= \frac{1}{2^n} \sum_x (f(x) - f(x^{\oplus i})) \chi_S(x) \\ &= \frac{1}{2^n} \sum_x f(x) \chi_S(x) - f(x^{\oplus i}) \chi_S(x) \end{aligned}$$

Because we are summing over all  $x$ , we can exchange  $f(x^{\oplus i})\chi_S(x)$  for  $f(x)\chi_S(x^{\oplus i})$ , arriving at,

$$\begin{aligned} \frac{1}{2^n} \sum_x f(x)\chi_S(x) - f(x^{\oplus i})\chi_S(x) &= \frac{1}{2^n} \sum_x f(x)\chi_S(x) - f(x)\chi_S(x^{\oplus i}) \\ &= \frac{1}{2^n} \sum_x f(x)(\chi_S(x) - \chi_S(x^{\oplus i})) \end{aligned}$$

At this point, we examine two scenarios.

1. If  $i$ , the flipped variable, is in  $S$ , then,

$$\chi_S(x) - \chi_S(x^{\oplus i}) = 2\chi_S(x)$$

whereby, we get,

$$\frac{1}{2^n} \sum_x 2f(x)\chi_S(x) = 2\hat{f}(S)$$

2. Conversely, If  $i$  is *not* in  $S$ , then,

$$\chi_S(x) - \chi_S(x^{\oplus i}) = 0$$

whereby, we get,

$$\frac{1}{2^n} \sum_x f(x)(0) = 0$$

Thus, we have shown that the influence of a variable  $i$  on a boolean function  $f$  is equal to the sum of the Fourier coefficients, squared, for any set that contains  $i$ .

$$\text{Inf}_i(f) = \sum_{S:i \in S} \hat{f}(S)^2$$

■

**Corollary 1** *Given a boolean function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ , the overall influence of  $f$  is computed as,*

$$\text{Inf}(f) = \sum_S |S| \hat{f}(S)^2$$

*Note that consequently, if  $f$  is balanced between outputs  $\{-1, 1\}$ , i.e.  $E[f] = \hat{f}(\emptyset) = 0$ , then  $\text{Inf}(f) \geq 1$ .*

## 2.2 Monotonicity

Moving on to the topic of monotonicity, we state the following definition.

**Definition 3** *A function is monotonically increasing if a “bigger” input results in a “bigger” output. Equivalently, a “bigger” input can never result in a “smaller” output.*

*For the class of boolean functions, we define monotonicity as follows. Let  $x, y$  denote binary strings,  $x, y \in \{-1, 1\}^n$ . We say that  $y \geq x$  if, for all  $i$ ,  $y_i \geq x_i$ . A boolean function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$  is monotone if*

$$y \geq x \implies f(y) \geq f(x)$$

*In other words, flipping bits from  $-1$  to  $1$  never causes the output to flip from  $1$  to  $-1$ .*

### Influence and Fourier concentration of monotonic functions

For monotone functions, there is a nice connection between the influence of a function and its Fourier coefficients.

**Lemma 3** *Given a monotone boolean function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ , we have*

$$\text{Inf}_i(f) = \hat{f}(\{i\}).$$

**Proof:** Let us consider the case where  $i = 1$ . Let  $x'$  denote a string of variables  $(x_2, \dots, x_n)$ , and let  $1x'$  denote the concatenation of  $1$  with  $x'$ , resulting in  $(1, x_2, \dots, x_n)$ . Thus, we have that,

$$\begin{aligned} \text{Inf}_i(f) &= \Pr_{x' \in \{-1, 1\}^{n-1}} [f(1x') \neq f(-1x')] \\ &= \frac{1}{2^{n-1}} (\text{number of } x' \text{ s.t. } f(1x') = 1, f(-1x') = -1) \end{aligned}$$

Recalling the definition of a Fourier coefficient, we have,

$$\begin{aligned} \hat{f}(\{i\}) &= E[f(x)x_i] \\ &= \frac{1}{2^n} \sum_{x' \in \{-1, 1\}^{n-1}} f(1x') - f(-1x') \end{aligned}$$

Note that the equation  $f(1x') - f(-1x')$  will never evaluate to  $-2$  because of monotonicity;  $0$  and  $2$  are the only possible values. Therefore, the equation of the Fourier coefficient can be rewritten as,

$$\begin{aligned} \hat{f}(\{i\}) &= \frac{1}{2^n} \sum_{x' \in \{-1, 1\}^{n-1}} f(1x') - f(-1x') \\ &= \frac{2}{2^n} (\text{number of } x' \text{ s.t. } f(1x') = 1, f(-1x') = -1) \\ &= \frac{1}{2^{n-1}} (\text{number of } x' \text{ s.t. } f(1x') = 1, f(-1x') = -1) \\ &= \text{Inf}_i(f) \end{aligned}$$

■

We can use this lemma to prove the following theorem.

**Theorem 1** For any monotone boolean function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ , we have

$$\text{Inf}(f) \leq \text{Inf}(\text{MAJ}) \leq \sqrt{n}.$$

**Proof:**

1. First, we must prove that the majority function has the greatest influence of any monotone function. It is given that  $f$  is monotone, therefore, by definition,

$$\text{Inf}(f) = \sum_{i=1}^n \text{Inf}_i(f)$$

Using the previous lemma, we have,

$$\begin{aligned} \text{Inf}(f) &= \sum_{i=1}^n \text{Inf}_i(f) \\ &= \sum_{i=1}^n \hat{f}(\{i\}) \\ &= \sum_{i=1}^n E[f(x)x_i] \\ &= E[f(x) \sum_{i=1}^n x_i] \\ &= \frac{1}{2^n} \sum_x f(x)(x_1 + \dots + x_n) \end{aligned}$$

The  $\pm 1$ -valued function  $f$  that maximizes this sum is obtained by taking  $f(x)$  to always equal  $\text{sign}(x_1 + \dots + x_n)$  (so each addend is nonnegative). This is the majority function. Therefore, the monotone function with maximum influence is the majority function.

2. We now wish to show that,

$$\sum_i \hat{f}(\{i\}) \leq \sqrt{n}$$

We know from Parseval's identity that

$$\sum_i \hat{f}(\{i\})^2 \leq 1.$$

Therefore, we can use Cauchy-Schwartz, which states that,

$$|\vec{u} \cdot \vec{v}| \leq \|\vec{u}\| \cdot \|\vec{v}\|$$

Thus, we have,

$$\begin{aligned} \sum_i \hat{f}(\{i\}) \cdot 1 &\leq \sqrt{\sum_i \hat{f}(\{i\})^2} \cdot \sqrt{\sum_i 1^2} \\ &\leq 1 \cdot \sqrt{n} = \sqrt{n} \end{aligned}$$

■

This theorem is useful in proving an upper bound on the Fourier coefficients of monotonic functions.

**Theorem 2** *Given a monotone boolean function  $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$ , the Fourier concentration of sets of cardinality  $\geq \left(\frac{\text{Inf}(f)}{\epsilon}\right)$  is  $\leq \epsilon$ .*

$$\sum_{|S| \geq \left(\frac{\text{Inf}(f)}{\epsilon}\right)} \hat{f}(S)^2 \leq \epsilon$$

**Proof:** We will proceed with a proof by refutation. As such, we assume the converse of the main theorem, namely that,

$$\sum_{|S| \geq \left(\frac{\text{Inf}(f)}{\epsilon}\right)} \hat{f}(S)^2 > \epsilon$$

If this is so, then,

$$\begin{aligned} \text{Inf}(f) &= \sum_{S \subseteq [n]} |S| \hat{f}(S)^2 \\ &\geq \sum_{|S| \geq \left(\frac{\text{Inf}(f)}{\epsilon}\right)} |S| \hat{f}(S)^2 \\ &\geq \left(\frac{\text{Inf}(f)}{\epsilon}\right) \sum_{|S| \geq \left(\frac{\text{Inf}(f)}{\epsilon}\right)} \hat{f}(S)^2 \end{aligned}$$

By the assumption,  $\sum_{|S| \geq \left(\frac{\text{Inf}(f)}{\epsilon}\right)} \hat{f}(S)^2 > \epsilon$ . Therefore, we would have,

$$\begin{aligned} \text{Inf}(f) &> \frac{\text{Inf}(f)}{\epsilon} \epsilon = \text{Inf}(f) \\ &\rightarrow \leftarrow \text{CONTRADICTION} \end{aligned}$$

Thus, since we have proven the assumption to be false, we can infer that the original theorem is true. ■

## Upper bounds

So far, we have proven that:

- Every monotone function  $f$  has overall influence  $\text{Inf}(f) \leq \sqrt{n}$ .
- Every monotone function  $f$  has Fourier concentration  $\sum_{|S| \geq \frac{\sqrt{n}}{\epsilon}} \hat{f}(S)^2 \leq \epsilon$ .

Thus, we have the following corollary.

**Corollary 2** *The Low-Degree Algorithm can learn any monotone function  $f$  to accuracy  $1 - \epsilon$  in  $n^{\frac{\sqrt{n}}{\epsilon}} = 2^{\frac{\sqrt{n} \log n}{\epsilon}}$ -time.*

**Open Question 2** *Is there a combinatorial algorithm to learn monotone functions in  $2^{\frac{\sqrt{n} \log n}{\epsilon}}$ -time?*

### Lower bounds

The class of monotonic functions is very rich. We will now show that most monotonic functions require boolean circuits of size  $\approx \frac{2^n}{\text{poly}(n)}$ . This counting argument can be used to prove a lower bound for learning monotone functions.

**Theorem 3** *Let  $A$  be any uniform distribution PAC learning algorithm for monotone functions on  $n$  variables, such that  $A$  runs in  $2^{0.9n}$ -time. Then  $A$  must sometimes output a hypothesis  $h$  with  $\text{error}(h) = \Omega\left(\frac{1}{\sqrt{n \log n}}\right)$ .*

**Proof:** Since  $A$  runs in  $\leq 2^{0.9n}$  many steps,  $A$  can output at most  $2^{2^{0.9n}}$  many distinct hypotheses. Let

$$m(n) = 2^{\binom{n}{n/2}}$$

Recall that,

$$\begin{aligned} \binom{n}{n/2} &\geq \frac{2^n}{2\sqrt{n}} \\ \implies m(n) &\geq 2^{\frac{2^n}{2\sqrt{2}}} \end{aligned}$$

There are at least  $m(n)$  many monotone functions (since any Boolean function that outputs 1 on all inputs with  $i > n/2$  ones and outputs 0 on all inputs with  $i < n/2$  ones is monotone, and there are  $m(n)$  functions of this sort). Therefore, there are  $\geq 2^{\frac{2^n}{2\sqrt{2}}}$  monotone functions.

Fix some hypothesis  $h$ .  $h$  is accurate for at most  $k(n) = \sum_{i=0}^{\epsilon 2^n} \binom{2^n}{i}$  functions (since this is the number of functions that disagree with  $h$  on  $0 \leq j \leq \epsilon 2^n$  many inputs). Using the standard inequality  $\sum_{i=0}^{\ell} \binom{N}{i} \leq (eN/\ell)^\ell$ , one obtains

$$k(n) \leq 2^{2^n \epsilon \log\left(\frac{e}{\epsilon}\right)}$$

If  $A$  can truly approximate any monotone function, then it must be the case that

$$2^{2^{0.9n}} k(n) \geq m(n)$$

This means it must be the case that

$$2^{2^{0.9n}} 2^{2^n \epsilon \log\left(\frac{e}{\epsilon}\right)} = 2^{2^{0.9n} + 2^n \epsilon \log\left(\frac{e}{\epsilon}\right)} \geq 2^{\frac{2^n}{2\sqrt{2}}}$$

Taking logs and doing some standard manipulations, it is straightforward to check that this inequality holds only if  $\epsilon \geq \Omega\left(\frac{1}{\sqrt{n \log n}}\right)$ . ■