COMS 6998-2 Advanced Topics in Computational Learning Theory	Spring 2005
Lecture 8 : March 10, 2005	
Lecturer: Rocco A. Servedio	*Scribe: Ariel Elbaz & Spyridon Antonakopoulos*

Overview of today's lecture

- "Low-degree" algorithm for learning via the Fourier transformation;
- Learning functions of halfspaces;
- **Readings:**
  1. Mansour, *Learning Boolean Functions via the Fourier Transform* ([1]);
  2. Klivans, O'Donnell and Servedio, *Learning intersections and thresholds of halfspaces* ([2]).

1 The Low-degree Algorithm

In this section we develop an efficient learning algorithm for learning Fourier coefficients $\hat{f}(S)$ of χ_S for sets $|S| \leq d$ and show why this algorithm is useful.

1.1 Preliminaries

Parseval's equality Let $f\colon \{1,-1\}^n \to \mathbb{R}$ be a function, and let $f(x) = \sum_{S\subseteq[n]} \hat{f}(S)\chi_S$ be the representation of f with Fourier coefficients. Recall Parseval's equality from last lecture:

$$\sum_{S\subseteq[n]} \hat{f}(S)^2 = \frac{1}{2^n} \sum_{x\in\{1,-1\}^n} f(x)^2 .$$

Note: Recall from linear algebra, if we look at f as a vector in $\mathbb{R}^{2^n}$, this is the property that vector length is an invariant of the basis used for representing the vector.

Example Oracle $\mathsf{EX}(f)$ is a uniformly random oracle for f, i.e. on every invocation $\mathsf{EX}(f)$ outputs a pair $(x, f(x))$, where x is selected uniformly at random from $\{1,-1\}^n$. A schematic representation of $\mathsf{EX}(f)$ is given in Figure 1.

For any given set $S \subseteq [n]$, we can approximate $\hat{f}(S)$ in polynomial time using $\mathsf{EX}(f)$, as indicated by the following lemma.

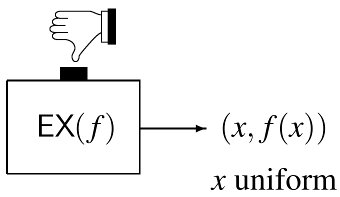

Figure 1: The EX(f) oracle

Lemma 1.1.1 (Efficient approximation of Fourier coefficients). *Let $S \subseteq [n]$. Given $\gamma > 0, \delta > 0$, and oracle access to* EX(f)*, there is an algorithm that outputs $c_S \in [-1, 1]$ such that $\left|c_S - \hat{f}(S)\right| \leq \gamma$ with probability at least $1 - \delta$, and runs in time $O\left(\frac{n}{\gamma^2} \log \frac{1}{\delta}\right)$.*

Proof. Recall that $\hat{f}(S) = \mathbf{E}_{x \in \mathcal{U}}[f(x)\chi_S] = 2\,\mathbf{Pr}[f(x) = \chi_S] - 1$. Therefore, it suffices to estimate $\mathbf{Pr}[f(x) = \chi_S]$ to accuracy $\pm\frac{\gamma}{2}$. The algorithm draws m (the value of m will be determined soon) random examples from EX(f) and estimates, based on these examples, the value of $\mathbf{Pr}[f(x) = \chi_S]$. Let Z_i be a Boolean random variable such that $Z_i = 1$ iff $f(x_i) = \chi_S(x_i)$. We know that $\mathbf{Pr}[Z_i = 1] = \hat{f}(S)$. Let $Z = \sum_{i=1}^{m} Z_i$, and let $\mu = \mathbf{E}[Z] = m\hat{f}(S)$. We can now use additive Chernoff bounds:

$$\begin{aligned}
\mathbf{Pr}\left[\left|\frac{Z}{m} - \hat{f}(S)\right| > \frac{\gamma}{2}\right] &= \mathbf{Pr}\left[Z > \left(\hat{f}(S) + \frac{\gamma}{2}\right)m\right] + \mathbf{Pr}\left[Z < \left(\hat{f}(S) - \frac{\gamma}{2}\right)m\right] \leq \\
&\leq e^{-2m(\frac{\gamma}{2})^2} + e^{-2m(\frac{\gamma}{2})^2} = \\
&= 2e^{-2m(\frac{\gamma}{2})^2}\ .
\end{aligned}$$

Solving the equation $2e^{-2m(\frac{\gamma}{2})^2} = \delta$ for m, we get that $m = O\left(\frac{1}{\gamma^2} \log \frac{1}{\delta}\right)$. As each example may take $O(n)$ time to process, the running time is $O\left(\frac{n}{\gamma^2} \log \frac{1}{\delta}\right)$. $\square$

Our next goal is to find "heavy" Fourier coefficients, when given access to EX(f). Suppose, for example, that there exists a set $S \subseteq [n]$ such that $\hat{f}(S)^2 = 99/100$. The crucial problem is whether we can determine S, given only access to EX(f), by an efficient algorithm. Unfortunately, this is an open problem, with reasonable probability that the answer may be negative.

Remark: We note here that in other models of learning the situation is very different. In particular, if the oracle for the function f accepts membership queries (i.e. takes as input any x of our choice and outputs $f(x)$) in addition to the standard functionality of EX(f), then it is possible to efficiently determine the sets with heavy Fourier coefficients.

However, not all hope is lost. As we have seen, we can approximate the value of $\hat{f}(S')$ for specific sets S' and thus we can do exhaustive search to find those with heavy Fourier coefficients. This "brute force" approach turns out to be useful, because for many boolean functions of interest almost all the Fourier coefficients correspond to sets of small cardinality. As a result, performing an exhaustive search on all sets S', such that $|S'| \leq d$ for a suitable value of d, will require approximating $\approx n^d$ Fourier coefficients, which is still tractable.

The Fourier Concentration measures the weight of the Fourier Coefficients that correspond to large sets.

Definition 1.1.2 (Fourier Concentration). *Given* $f\colon \{1,-1\}^n \to \{1,-1\}$*, we say that* f *has Fourier concentration* $\alpha(\varepsilon, n)$ *if* $\forall \varepsilon \in (0,1)$*, we have* $\sum_{|S|>\alpha(\varepsilon,n)} \hat{f}(S)^2 \leq \varepsilon$.

We shall now give an example of the use of this definition.

Proposition 1.1.3. *Any AND function* f *of some set of* k *variables has Fourier concentration of* $\log \frac{4}{\varepsilon}$.

Proof. Consider the two cases below:

1. If $k \leq \log \frac{4}{\varepsilon}$, then for any set $S \subseteq [n]$ such that $|S| > \log \frac{4}{\varepsilon}$, S contains at least one variable which is not in the AND. Hence, it is easy to verify that $\hat{f}(S) = 0$.

2. If $k > \log \frac{4}{\varepsilon}$, we recall from the previous lecture that, for every set $S \neq \emptyset$ which contains only variables that appear in the AND, $\hat{f}(S)$ has value $\pm \frac{2}{2^k}$. Thus, the sum $\sum_{|S|>0} \hat{f}(S)^2$ does not exceed $2^k \cdot \left(\frac{2}{2^k}\right)^2 = \frac{4}{2^k}$. Since we assumed $k > \log \frac{4}{\varepsilon}$, this is at most ε.

Consequently, $\alpha(\varepsilon, n) \leq \log \frac{4}{\varepsilon}$ for such a function f. $\square$

1.2 Approximating functions with constant Fourier concentration

Suppose that f has Fourier concentration $d = \alpha(\varepsilon, n)$, independent of n. The "low-degree" algorithm $\mathcal{A}$ works as follows:

1. Accuracy and confidence parameters $\tau, \delta > 0$ are given as input, and oracle access to $\mathsf{EX}(f)$ is available.

2. $\mathcal{A}$ draws $m = O\left(\frac{n^d}{\tau} \cdot \log\left(\frac{n^d}{\delta}\right)\right)$ examples from $\mathsf{EX}(f)$ and uses them to approximate $\hat{f}(S)$, for each S such that $|S| \leq d$. Denote by c_S the estimated value for $\hat{f}(S)$.

3. $\mathcal{A}$ outputs the hypothesis $h(x) = \sum_{|S|\leq d} c_S \chi_S$.

Theorem 1.2.1. *If* f *has Fourier concentration* $d = \alpha(\varepsilon, n)$*, then with probability at least* $1-\delta$ *the function* $h\colon \{1,-1\}^n \to \mathbb{R}$ *outputted by algorithm* $\mathcal{A}$ *has* $\mathbf{E}_{x\in\mathcal{U}}\left[(h(x) - f(x))^2\right] \leq \varepsilon + \tau$.

Proof. Consider some specific $S \subseteq [n]$ such that $|S| \le d$ and let $\gamma = \sqrt{\frac{\tau}{n^d}}$. From Lemma 1.1.1, a sample of size m suffices to give a coefficient estimate c_S such that $\left|c_S - \hat{f}(S)\right| \le \gamma$ with probability at least $1 - \frac{\delta}{n^d}$. By union bound, we get that the probability that there exists a set $|S| \le d$ such that $\left|c_S - \hat{f}(S)\right| > \gamma$ is at most $n^d \cdot \frac{\delta}{n^d} = \delta$. Thus, we assume that with probability at least $1-\delta$, all estimated coefficients c_S are at least γ-close to the Fourier coefficients $\hat{f}(S)$.

Let $g(x) = f(x) - h(x)$ be the difference function between f and h. Due to linearity, for every set S we have $\hat{g}(S) = \hat{f}(S) - \hat{h}(S)$. When $|S| \le d$, we know that $\hat{h}(S) = c_S$ and when $|S| > d$, $\hat{h}(S) = 0$. By Parseval's Theorem, $\mathbf{E}\left[(h(x) - f(x))^2\right] = \mathbf{E}\left[g(x)^2\right] = \mathbf{E}\left[\hat{g}(S)^2\right]$. We expand to get

$$
\begin{aligned}
\mathbf{E}\left[\hat{g}(S)^2\right] &= \sum_{|S|>d} \hat{g}(S)^2 + \sum_{|S|\le d} \hat{g}(S)^2 \\
&= \sum_{|S|>d} \hat{f}(S)^2 + \sum_{|S|\le d} (\hat{f}(S) - c_S)^2 \\
\text{(with probability } 1-\delta) \quad &\le \varepsilon + \sum_{|S|\le d} \gamma^2 \\
&\le \varepsilon + \tau\,,
\end{aligned}
$$

thus proving the desired result. □

The reason why the above bound on $\mathbf{E}\left[(h(x) - f(x))^2\right]$ is useful will be elucidated below.

Theorem 1.2.2. *Given a boolean function* $f\colon \{1,-1\}^n \to \{1,-1\}$ *and a real function* $h\colon \{1,-1\}^n \to \mathbb{R}$, *then*

$$\Pr_x[f(x) \ne \operatorname{sign}(h(x))] \le \mathop{\mathbf{E}}_x\left[(f(x) - h(x))^2\right] .$$

Proof. We write the right-hand side as

$$\mathop{\mathbf{E}}_x\left[(f(x) - h(x))^2\right] = \frac{1}{2^n} \sum_x \left[(f(x) - h(x))^2\right] \tag{1}$$

and the left-hand side as

$$\Pr_x[f(x) \ne \operatorname{sign}(h(x))] = \frac{1}{2^n}\left[\sum_{f(x)\ne\operatorname{sign}(h(x))} 1 \quad + \sum_{f(x)=\operatorname{sign}(h(x))} 0\right] . \tag{2}$$

We claim that every x contributes more to (1) than to (2). Indeed, if x is such that $f(x) = sign(h(x))$, then x does not contribute to (2), but it may contribute some positive value to (1), in case $f(x)$ and $h(x)$ are not equal. On the other hand, if x is such that $f(x) \ne sign(h(x))$, then $|f(x) - h(x)| \ge 1$, because f is a boolean function. So, the contribution of x to (1), which is $(f(x) - h(x))^2$, equals or exceeds the contribution of x to (2), which is 1. □

It remains to combine the two results of Theorems 1.2.1 and 1.2.2:

Theorem 1.2.3. *Let $\mathcal{C}$ be a concept class such that every $f \in \mathcal{C}$ has Fourier concentration $d = \alpha(\varepsilon, n)$; then, there is a PAC learning algorithm for $\mathcal{C}$ that runs in time $\frac{n^{O(d)}}{\varepsilon} \log\left(\frac{n^d}{\delta}\right)$.*

Proof. Run the aforementioned low-degree algorithm with $\tau = \varepsilon$, and output $\mathrm{sign}(h(x))$ as hypothesis. By Theorems 1.2.1 and 1.2.2, the error percentage will be

$$\Pr_x[f(x) \neq \mathrm{sign}(h(x))] \leq \mathop{\mathbf{E}}_x\left[(f(x) - h(x))^2\right] \leq \varepsilon + \tau = 2\varepsilon\,.$$

Furthermore, recall that the low-degree algorithm requires $m = O\left(\frac{n^d}{\tau} \cdot \log\left(\frac{n^d}{\delta}\right)\right)$ samples and hence it runs in time $\frac{n^{O(d)}}{\varepsilon} \log\left(\frac{n^d}{\delta}\right)$. □

The above theorems allow us to analyse a function by focusing on its Fourier concentration, instead of dealing with the "logical structure" of the function.

2 Learning Halfspaces

2.1 Overview of results

The next theorem bounds the Fourier concentration of halfspaces. Observe that it holds for *any* halfspace, and does not depend on the weight of $\vec{w}$ or the margin of the examples.

Theorem 2.1.1. *For any halfspace $f(x) = sign(\vec{w} \cdot \vec{x} - \theta)$, we have*

$$\sum_{|S| > \frac{441}{\varepsilon^2}} \hat{f}(S)^2 \leq \varepsilon\,.$$

Corollary 2.1.2. *We can learn halfspaces in $n^{O\left(\frac{1}{\varepsilon^2}\right)}$-time.*

It should be noted that this time bound is much worse than previously achieved results (cf. Winnow). The importance of this corollary will come to view later in the section.

Similarly, we have a bound on the Fourier concentration of arbitraty functions on k halfspaces:

Theorem 2.1.3. *Let f be any function of k halfspaces, i.e. $f(x) = g(h_1, \ldots, h_k)$, where $g\colon \{1,-1\}^k \to \{1,-1\}$ and $h_1, h_2, \ldots, h_k$ are halfspaces. Then*

$$\sum_{|S| > \frac{441k^2}{\varepsilon^2}} \hat{f}(S)^2 \leq \varepsilon\,.$$

Corollary 2.1.4. *We can learn such functions f on k halfspaces in $n^{O\left(\frac{k^2}{\varepsilon^2}\right)}$-time.*

2.2 Noise sensitivity of Boolean functions

We start with a few definitions:

Definition 2.2.1 (Noise Operator). *Given noise rate* ε, $0 < \varepsilon < 1/2$, *the noise operator* $N_\varepsilon \colon \{1,-1\}^n \to \{1,-1\}^n$ *flips each bit with probability* ε. *That is, if* $y = N_\varepsilon(x)$ *we have that for every* $j \in [n]$, $\mathbf{Pr}\big[y_j \neq x_j\big] = \varepsilon$.

Definition 2.2.2 (Noise Sensitivity). *The noise sensitivity of* $f \colon \{1,-1\}^n \to \{1,-1\}$ *at a noise rate* ε *is*

$$ns_\varepsilon(f) \triangleq \mathop{\mathbf{Pr}}_{x,\text{noise}}[f(x) \neq f(N_\varepsilon(x)] \,.$$

A few examples of the noise sensitivity of different functions:

- $f(x) = x_1$:
  It is easy to see that $ns_\varepsilon(f) = \varepsilon$.

- $f(x) = x_1 \wedge x_2 \wedge \cdots \wedge x_k$:
  Note that when x is uniformly distributed, $N_\varepsilon(x)$ is also uniformly distributed (although, of course, x and $N_\varepsilon(x)$ are not independently distributed). We thus have

$$\begin{aligned} ns_\varepsilon(f) &= \mathop{\mathbf{Pr}}_x[f(x) \neq f(N_\varepsilon(x))] = \\ &= \mathop{\mathbf{Pr}}_x[(f(x) = 1) \text{ and } (f(N_\varepsilon(x)) = 0)] + \\ &\qquad + \mathop{\mathbf{Pr}}_x[(f(x) = 0) \text{ and } (f(N_\varepsilon(x) = 1)] = \\ \text{(due to symmetry)} \quad &= 2\mathop{\mathbf{Pr}}_x[(f(x) = 1) \text{ and } (f(N_\varepsilon(x)) = 0)] = \\ &= 2 \cdot \frac{1}{2^k} \cdot \mathop{\mathbf{Pr}}_x[\text{ any of the bits } 1,\ldots,k \text{ is flipped }] = \\ &= 2 \cdot \frac{1}{2^k} \cdot \left(1 - (1-\varepsilon)^k\right). \end{aligned}$$

- $f(x) = \mathsf{MAJ}_n(x) = \text{sign}(x_1 + \cdots + x_n)$:

  Proposition 2.2.3. $ns_\varepsilon(\mathsf{MAJ}_n) = O(\sqrt{\varepsilon})$.

  Sketch of proof. It must be stressed that, for the purposes of this class, we shall present here an intuitive argument for the aforementioned proposition, *not* a rigorous proof.

  Suppose that the values of the components of the random vector $x \in \{1,-1\}^n$ are given one by one, such that x_i becomes known at time i. Now consider the partial sum $\Sigma_i(x) = x_1 + \cdots + x_i$: as i increases, $\Sigma_i(x)$ behaves like an unbiased random walk $\mathcal{R}$ on the number line. The absolute value of this sum after n steps is expected to be roughly

$$|\Sigma_n(x)| \approx \sqrt{n} \,.$$

Recall that the noise operator N_ε flips each bit of x with probability ε, so the expected number of flips is εn. In a manner similar to the above, we can think of the effect of the noise operator N_ε (when applied to x) on $\Sigma_n(x)$ as another random walk $\mathcal{Q}$, consisting of εn steps and beginning at the point $\Sigma_n(x)$, i.e. where $\mathcal{R}$ stopped. We also make the simplifying assumption that a bit flipped by noise has more or less equal probability of originally being either 1 or -1, which implies that $\mathcal{Q}$ is unbiased.

The expected absolute displacement of $\mathcal{Q}$, with respect to its starting point, is $\sqrt{\varepsilon n}$. So, by Markov's inequlity, the probability that this quantity will exceed $\sqrt{n}$ is $\leq \sqrt{\varepsilon}$. However, observe that for $\Sigma_n(N_\varepsilon(x))$ to have a different sign from $\Sigma_n(x)$, the absolute displacement of $\mathcal{Q}$ must be, on average, greater than $\sqrt{n}$. Consequently,

$$\begin{aligned}\mathbf{Pr}[\mathsf{MAJ}_n(N_\varepsilon(x)) \neq \mathsf{MAJ}_n(x)] &= \mathbf{Pr}[\operatorname{sign}(S_n(N_\varepsilon(x))) \neq \operatorname{sign}(S_n(x))] = \\ &= \mathbf{Pr}\big[\text{ absolute displacement of } \mathcal{Q} \text{ is} > \sqrt{n}\,\big] \leq \\ &\leq \sqrt{\varepsilon}\,,\end{aligned}$$

whence the proposition follows. $\square$

In fact, there is a more general result:

Fact 2.2.4 ([3]). *For* any *halfspace* $h(x) = \operatorname{sign}(\vec{w}\cdot\vec{x} - \theta)$, *the noise sensitivity* $ns_\varepsilon(h) = O(\sqrt{\varepsilon})$.

- $f(x) = \mathsf{PAR}(x_1, \ldots, x_k)$.

Proposition 2.2.5. *The noise sensitivity* $ns_\varepsilon(f) = \frac{1}{2}\big(1-(2-2\varepsilon)^k\big)$.

Proof. The noise changes the value of f if an odd number of variables $x_1, \ldots, x_k$ are flipped:

$$ns_\varepsilon(f) = \mathbf{Pr}[\text{ odd \# bits flipped }] = \sum_{j \text{ odd}} \binom{k}{j} (1-\varepsilon)^{k-j+1}\varepsilon^j\,.$$

To get the desired result, we look at the binomial expansions of $1^k = ((1-\varepsilon)+\varepsilon)^k$ and $(1-2\varepsilon)^k = ((1-\varepsilon)-\varepsilon)^k$:

$$((1-\varepsilon)+\varepsilon)^k = \sum_{j=0}^{k} \binom{k}{j} (1-\varepsilon)^{k-j+1}\varepsilon^j$$

and

$$((1-\varepsilon)-\varepsilon)^k = \sum_{j=0}^{k} (-1)^j \cdot \binom{k}{j} (1-\varepsilon)^{k-j+1}\varepsilon^j\,.$$

Observe that

$$\frac{1}{2}\big(1^k - (1-2\varepsilon)^k\big) = \sum_{j \text{ odd}} \binom{k}{j} (1-\varepsilon)^{k-j+1}\varepsilon^j = ns_\varepsilon(f)\,,$$

which we sought to prove. $\square$

2.3 Noise sensitivity vs. Fourier representation

Theorem 2.3.1. *The noise sensitivity $ns_\varepsilon(f)$ is equal to*

$$\frac{1}{2}\Big(1-\sum_{S\subseteq[n]}(1-2\varepsilon)^{|S|}\hat{f}(S)^2\Big)\,.$$

For small sets S, the value of $(1-2\varepsilon)^{|S|}$ is relatively large. In other words, if a function f is concentrated on low degree sets, we can expect the noise sensitivity to be small.

Proof. Let $x\in\{1,-1\}^n$ be a uniformly selected random variable. Recall that $y=N_\varepsilon(x)$ will also be uniformly distributed. We write the noise sensitivity in terms of the joint probability distribution of x and y:

$$\begin{aligned}
ns_\varepsilon(f) &= \mathop{\mathbf{Pr}}_{x,y}[f(x)\neq f(y)] = 1-\mathop{\mathbf{Pr}}_{x,y}[f(x)=f(y)] = \\
\text{(using indicator random variable)}\quad &= 1-\mathop{\mathbf{E}}_{x,y}\big[1-1_{f(x)=f(y)}\big] = \\
&= 1-\mathop{\mathbf{E}}_{x,y}\left[1-\frac{(f(x)-f(y))^2}{4}\right] = \\
\text{(due to linearity of expectation)}\quad &= \frac{1}{4}\mathop{\mathbf{E}}_{x,y}\big[(f(x)-f(y))^2\big] = \\
&= \frac{1}{4}\mathop{\mathbf{E}}_{x,y}\big[f(x)^2+f(y)^2-2f(x)f(y)\big] = \\
&= \frac{1}{4}\left(2-\mathop{\mathbf{E}}_{x,y}[f(x)f(y)]\right)\,.
\end{aligned}$$

We now wish to show that $\mathbf{E}_{x,y}[f(x)f(y)]=\sum_{S\subseteq[n]}(1-2\varepsilon)^{|S|}\hat{f}(S)^2$. We write $f(x)=\sum_S\hat{f}(S)x_S$, and $f(y)=\sum_T\hat{f}(T)y_T$, then expand to get

$$\begin{aligned}
\mathop{\mathbf{E}}_{x,y}[f(x)f(y)] &= \mathop{\mathbf{E}}_{x,y}\left[\Big(\sum_S\hat{f}(S)x_S\Big)\Big(\sum_T\hat{f}(T)y_T\Big)\right] = \\
&= \sum_{S,T}\hat{f}(S)\hat{f}(T)\;\mathop{\mathbf{E}}_{x,y}[x_Sy_T]\,.
\end{aligned}$$

If $S\neq T$, $\mathbf{E}_{x,y}[x_Sy_T]=0$. Indeed, assume that $T\setminus S\neq\emptyset$ and let $i\in T\setminus S$. Since y is uniformly distributed, $\mathbf{Pr}[y_i=1]=\mathbf{Pr}[y_i=-1]$, therefore

$$\begin{aligned}
\mathop{\mathbf{E}}_{x,y}[x_Sy_T] &= \frac{1}{2}\big(\mathbf{E}[x_Sy_T\mid y_i=1]+\mathbf{E}[x_Sy_T\mid y_i=-1]\big) = \\
&= \frac{1}{2}\big(\mathbf{E}\big[x_Sy_{T\setminus\{i\}}\big]-\mathbf{E}\big[x_Sy_{T\setminus\{i\}}\big]\big) = 0\,.
\end{aligned}$$

Hence, $\mathbf{E}_{x,y}[f(x)f(y)]=\sum_S\hat{f}(S)^2\,\mathbf{E}[x_Sy_S]$. Moreover, observe that

$$\begin{aligned}
\mathbf{E}[x_Sy_S] &= \mathbf{Pr}[x_S=y_S]-(1-\mathbf{Pr}[x_S=y_S]) = \\
&= 2\,\mathbf{Pr}[x_S=y_S]-1 =
\end{aligned}$$

$$
\begin{aligned}
&= 1 - 2\,\mathbf{Pr}[x_S \neq y_S] = \\
&= 1 - 2ns_\varepsilon(\mathsf{PAR}_S) = \\
&= 1 - 2\left(\frac{1}{2}\left(1-(1-2\varepsilon)^{|S|}\right)\right) = (1-2\varepsilon)^{|S|}\,,
\end{aligned}
$$

thus concluding the proof. □

References

[1] Y. Mansour, *Learning Boolean Functions via the Fourier Transform*. Theoretical Advances in Neural Computation and Learning, 391–424, 1994.

[2] A. Klivans, R. O'Donnell, R. Servedio, *Learning intersections and thresholds of halfspaces*. In Proc. 43rd Ann. IEEE Symp. on Foundations of Comp. Sci.,pages 177–186, 2002.

[3] Y. Peres, private communication.