

## Lecture 18

- Learning heavy Fourier coeffs (with queries)  
(cont.)
- weak learning of monotone fctns

begin review:

Recall Fourier Transform:

$$\chi_s(x) = \prod_{i \in S} x_i$$

$$\langle f, g \rangle = \frac{1}{2^n} \sum_x f(x) g(x)$$

$$\hat{f}(s) \equiv \langle f, \chi_s \rangle = 1 - 2 \cdot \Pr[f(x) \neq \chi_s(x)]$$

lemma

$$= 2 \cdot \Pr[f(x) = \chi_s(x)] - 1$$

$$\forall f, f(x) = \sum_s \hat{f}(s) \chi_s(x)$$

$$\text{Plancherel } \langle f, g \rangle = \sum_s \hat{f}(s) \hat{g}(s)$$

Parseval's  
for  
Boolean

$$1 = \langle f, f \rangle = \sum_s \hat{f}(s)^2$$

Boolean  
↓  
Parseval  
↓

$$f: \{\pm 1\}^n \rightarrow \{\pm 1\}$$

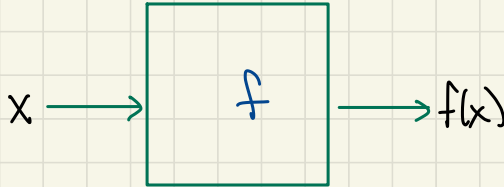
Can approx  $\hat{f}(s)$  to additive  $\epsilon$  with  $O\left(\frac{1}{\epsilon^2} \log \frac{1}{\delta}\right)$  Samples

Confidence  
↓

# Learning Heavy Fourier Coeffs

[Goldreich Levin]

[Kushilevitz Mansour]



not just low degree  $S$



all close linear fctns

Given  $f, \theta$

• Output all coeffs  $S$  st.  $|\hat{f}(S)| \geq \theta$

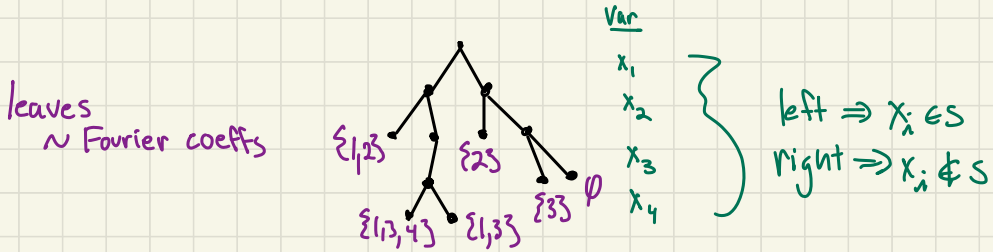
• Only output  $S$  st.  $|\hat{f}(S)| \geq \frac{\theta}{2}$  ← no junk

Probably can't do it with only random examples

What if can query  $f$  at any input?

Observe if  $\hat{f}(S) > 3/4$  can use self-correction to find it

# Main Idea: "exhaustive search with good pruning"



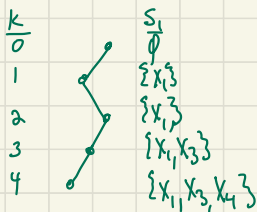
ONLY OUTPUT THOSE THAT REACH BOTTOM LEVEL

recursive algorithm:

- each node  $\sim$  setting of  $x_1, \dots, x_i$
- estimate "total energy" of subtrees  $x_1 \dots x_i (x_{i+1} = +1)$   
 $\& x_1 \dots x_i (x_{i+1} = -1)$
- only go down paths with high enough energy

How to prune?

Define quantity:



Fix  $0 \leq k \leq n$   
 $S_1 \subseteq [k]$

current "level" of search  
 current "node" of search

doesn't depend on 1st k bits  
same for all

$2^k$   
such  
fctns  
(for each  
 $s_1$ )

$$f_{k, s_1}: \{\pm 1\}^{n-k} \rightarrow \mathbb{R}$$

all Fourier coeffs  
which agree on first k  
elements

$$\text{s.t. } f_{k, s_1}(x) = \sum_{T_2 \subseteq \{k+1, \dots, n\}} \hat{f}(s_1, T_2) \chi_{T_2}(x)$$

all extensions  
of  $s_1$  to indices  
in  $\{k+1, \dots, n\}$

could be  
 $s_1, T_2$   
but no need  
since  $\chi_{s_1, T_2} = \chi_{s_1} \cdot \chi_{T_2}$   
same for  
all

notation:  
index 1  $\rightarrow$  prefix  
2  $\rightarrow$  suffix

Sanity Checks:

1)  $k=0$

$$f_{0, \emptyset}(x) = \sum_{T_2 \subseteq [n]} \hat{f}(T_2) \chi_{T_2}(x) = f(x)$$

$\uparrow$  since  $k=0$   
 $\uparrow$  since  $s_1 = \emptyset$

2)  $k=n$

$$f_{n, s_1}(x) = \hat{f}(s_1) \cdot \chi_{\emptyset}(x)$$

$\uparrow$  sum over  $T_2 = \emptyset$   
 $\uparrow$  since  $T_2 = \emptyset$   
 $\overbrace{\chi_{\emptyset}(x)} = 1$

Plan Only go down paths with  $E[f^2(x)] \geq \theta^2$   
 $K_S$

1. can we compute it?

2. does it bring us to right leaves?

- do we get to all heavy leaves?

- do we get junk? (light leaves)

3. how many paths do we take?

lots of dead ends?

is runtime good?

# Not too many paths! (answer to 3)

Lemma "not too many" ← at any stage in algorithm

$$f: \{\pm 1\}^n \rightarrow \{\pm 1\}$$

$$(1) \leq \frac{1}{\theta^2} \text{ } s_1 \text{ 's satisfy } |\hat{f}(s_1)| \geq \theta$$

$$(2) \forall 0 \leq k \leq n, \leq \frac{1}{\theta^2} \text{ fctns } f_{k, s_1}$$

$$\text{have } E_x[f_{k, s_1}^2] \geq \theta^2$$

(Proved last lecture)

Useful claim (proved last time):

Claim:  $\forall k, s_1 \leq k$

$$E_x[f_{k, s_1}(x)^2] = \sum_{T_2 \subseteq \{k+1, \dots, n\}} \hat{f}(s_1 \cup T_2)^2$$

(end review)

Does algorithm bring us to good leaves?

(answer to 2)

Fact: "not missing out"  $\Rightarrow$  find all big Fourier coeffs  
For any  $S_1$ , if  $\exists T_2$  st.

$$|\hat{f}(S_1 \vee T_2)| > \theta$$

then  $E_x [f_{k_{S_1}}^2(x)] = \sum_{T_2} \hat{f}(S_1 \vee T_2)^2$  via claim  
 $\geq \theta^2$

$\Rightarrow E_x [f_{k_{S_1}}^2(x)]$  is a good measure  
to use when deciding whether  
to investigate subtree!

So we find all good leaves  
we don't spend too much time

but do we output junk?

No junk (answer to 2)

Simple fix:

For each "candidate" to  $S$ ,  
estimate its Fourier coeff & make sure it is  
big enough before outputting it

What's left?

Can we estimate  $f_{k, s_1}(x)$ ?

recall:

$$f: \{\pm 1\}^n \rightarrow \{\pm 1\}$$
$$0 \leq k \leq n$$
$$S_1 \subseteq [k]$$

(answer to 1)

$$f_{k, s_1}(x) = \sum_{T_2 \subseteq [k, \dots, n]} \hat{f}(S_1 \cup T_2) \chi_{T_2}(x)$$

$$E_x [f_{k, s_1}(x)^2] = \sum_{T_2 \subseteq [k, \dots, n]} \hat{f}(S_1 \cup T_2)^2$$

Bad idea: estimate each

$$\hat{f}(S_1 \cup T_2) \quad \forall T_2 \leftarrow \text{too much time}$$

Another challenge:

heavy  $\hat{f}(S_1 \cup T_2)$  can differ

greatly depending on how

you fix vars in  $S_1$   $\leftarrow$  should you set randomly?

e.g.  $S_1 = \{x_1\}$

$$f(x) = \begin{cases} 1 & \text{if } x_1 = 1 \\ \prod x_i & \text{if } x_1 = -1 \end{cases}$$

$\leftarrow$  all  $\hat{f}(s) = 0$   
for  $s \neq \emptyset$   
 $\hat{f}(\emptyset) = 1$

$$\hat{f}(s = [n]) = 1$$

$$\hat{f}(s) = 0 \quad \forall s \neq [n]$$

Can we estimate  $f_{k,s_1}(x)$ ?

recall:

$$f: \{\pm 1\}^n \rightarrow \{\pm 1\}$$

$$0 \leq k \leq n$$

$$s_1 \in [k]$$

(answer to 1)

note:  $x$  is suffix of original input  
 $y$  is prefix

$$f_{k,s_1}(x) = \sum_{T_2 \in \{k+1..n\}} \hat{f}(s_1, T_2) \chi_{T_2}(x)$$

$$E_x [f_{k,s_1}(x)^2] = \sum_{T_2 \in \{k+1..n\}} \hat{f}(s_1, T_2)^2$$

Lemma " $f_{k,s_1}(x)$  Estimation lemma"

for  $x \in \{\pm 1\}^{n-k}$

$$f_{k,s_1}(x) = E_{y \in \{\pm 1\}^k} [f(yx) \chi_{s_1}(y)]$$

} Use this to estimate  $f_{k,s_1}(x)$

settings of  
 1st  $k$  bits

concatenation

"agreement"

think of as  
 fctn of  $y$  since  
 $x$  is fixed  
 throughout

pf  
 Fourier representation  $\Rightarrow$

$$f(yx) = \sum_T \hat{f}(T) \chi_T(yx)$$

$$T = T_1 \cup T_2 \quad T_1 \in [k] \quad T_2 \in \{k+1..n\}$$

$$\text{so } \chi_T(yx) = \chi_{T_1}(y) \cdot \chi_{T_2}(x)$$

$$E_y[f(y|x)\chi_{S_1}(y)]$$

$$= E_y\left[\sum_{T_1} \sum_{T_2} \hat{f}(T_1, T_2) \chi_{T_1}(y) \chi_{T_2}(x) \cdot \chi_{S_1}(y)\right]$$

by  
above

$$= \sum_{T_1} \sum_{T_2} \hat{f}(T_1, T_2) \chi_{T_2}(x) E_y[\chi_{T_1}(y) \chi_{S_1}(y)]$$

$$= \begin{cases} 0 & \text{if } T_1 \neq S_1 \\ 1 & \text{if } T_1 = S_1 \end{cases}$$

$$= \sum_{T_2} \hat{f}(S_1, T_2) \chi_{T_2}(x)$$

$$= f_{k, S_1}(x)$$



An algorithm to estimate  $E_x [f_{k,S_i}(x)^2]$ :

Pick random  $x$ 's  $\in \{\pm 1\}^{n-k}$

For each  $x$ , pick random  $y$ 's  $\in \{\pm 1\}^k$

Estimate  $E_{y_1, y_2} [f(y, x) \chi_{S_i}(y) f(y, x) \chi_{S_i}(y_2)]$

which gives estimate of  
 $f_{k,S_i}^2(x)$

Estimate  $E_x [f_{k,S_i}(x)^2]$  via average of above

Chernoff + samples

$\Rightarrow$  Can get  $\gamma$ -additive estimate with  
prob  $\geq 1 - \delta$  in  $O\left(\frac{1}{\gamma^2} \log \frac{1}{\delta}\right)$  queries

$\Rightarrow$  Can get  $\frac{\gamma^2}{2}$ -additive est of  $f_{k,S_i}^2(x)$

# Algorithm:

Ideal [KM] algorithm (given  $k, s_i$ ):

if  $k = n$

(\*) if  $\frac{\theta}{4}$ -additive-estimate of  $\hat{f}(s_i) \geq \frac{3}{4}\theta$  output  $s_i$

test  $s_i$  is indeed heavy

else (1) if  $E_x[f_{k, s_i \cup \{k+i\}}^2(x)] \geq \frac{\theta^2}{2}$

recurse on  $(k+i, s_i \cup \{k+i\})$

(else kill this subtree)

check out left subtree

(2) if  $E_x[f_{k, s_i}^2(x)] \geq \frac{\theta^2}{2}$

recurse on  $(k, s_i)$

(else kill this subtree)

check out right subtree

Thm  $\forall \theta > 0$ , KM-alg outputs

$S = \{s_1, \dots, s_\ell\}$  st.  $\ell = O(1/\theta^2)$  + with prob  $\geq 1 - \delta$

$$\forall s_i \in S \quad |\hat{f}(s_i)| \geq \frac{\theta}{2} \quad \text{no junk}$$

$$\forall s \notin S \quad |\hat{f}(s)| \leq \theta \quad \text{no misses}$$

+ query time is poly  $(n, \frac{1}{\theta}, \log \frac{1}{\delta})$

Pf. if  $\hat{f}(s) < \frac{\theta}{2}$ ,  $\hat{f}(s)^2 \leq \frac{\theta^2}{4}$

+ test  $\times$  prevents it from being output

if  $\hat{f}(s) > \theta$ ,  $\forall k$  if  $s_1, \dots, s_k$  agree on  $[1..k]$

"not missing out" fact  $\Rightarrow$

$$E_x [f_{k, s}^2(x)] \geq \theta^2$$

total # nodes explored at level  $k \leq \frac{1}{\theta^2}$

$$\Rightarrow \text{total nodes explored} \leq \frac{n}{\theta^2}$$

## Applications

- size  $\leq t$  decision trees
- fctns of small  $L_1$ -norm

$$L_1(f) = \sum_s |\hat{f}(s)|$$

$$\text{set } \theta \leftarrow \frac{\epsilon}{L_1(f)}$$

in time  $\text{poly}(n, L_1(f), 1/\epsilon)$

if don't know  $L_1(f)$

assume 1, 2, 4, 8

test hypothesis at each round

& continue if not good

# Monotone Functions

def. partial order  $\leq$ :  $x \leq y$  iff  $\forall i \ x_i \leq y_i$

monotone fctn  $f$ :  $x \leq y \Rightarrow f(x) \leq f(y)$

Are there fast learning algorithms for the class of monotone functions?

Occam's razor:

poly  $(\log |\mathcal{C}|)$  samples suffice

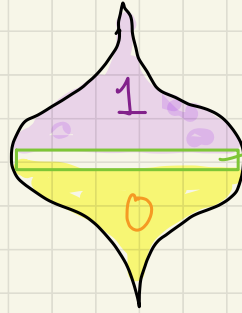
↑ class of monotone fctns

$\geq 2^{\frac{2}{\sqrt{n}}}$  monotone fctns

so only gives exponential bound

Why so many monotone fctns?

Consider "slice" fctns;



set middle row  
in all possible  
ways w/o  
violating monotonicity

$2^{\binom{n}{2}}$  options  
all are monotone!

H.W.:  $2^{O(\sqrt{n})}$  random samples suffice  
for unif dist

Today: what if you have queries?

can get very slight "win"

All monotone fctns have weak  
agreement with some dictator  
fctn.

Thm  $\forall f$  monotone,  $\exists g \in \{\pm 1, x_1, x_2, \dots, x_n\} \equiv S$

$$\text{s.t. } \Pr_x [f(x) = g(x)] \geq \frac{1}{2} + \Omega\left(\frac{1}{n}\right)$$

uniform distribution

slightly better than random guessing

note Slice fctns have weak agreement with all dictators on uniform dist

(can get  $\frac{1}{2} + \Omega\left(\frac{1}{n}\right)$  if add majority)

$\Rightarrow$  learning algorithm:

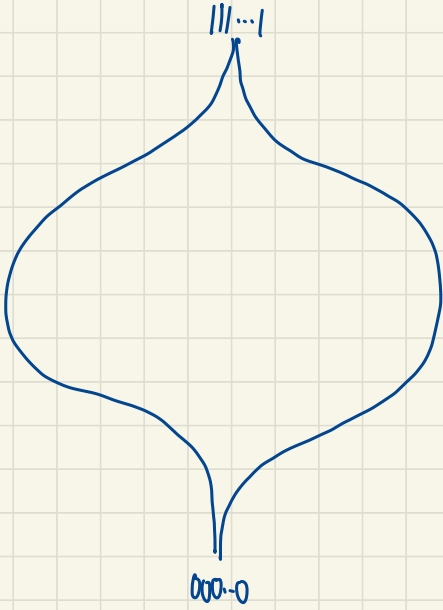
estimate agreement of  $f$  with all members of  $S'$   
output best

Pf.

Case 1:  $f(x)$  has weak agreement with  $+1$  or  $-1$  ✓

Case 2: otherwise  $\Pr[f(x)=1] \in \left[\frac{1}{4}, \frac{3}{4}\right]$

# Boolean Cube



hypercube

level  $k$ :

nodes labeled by

$k$  1's +  $n-k$  0's

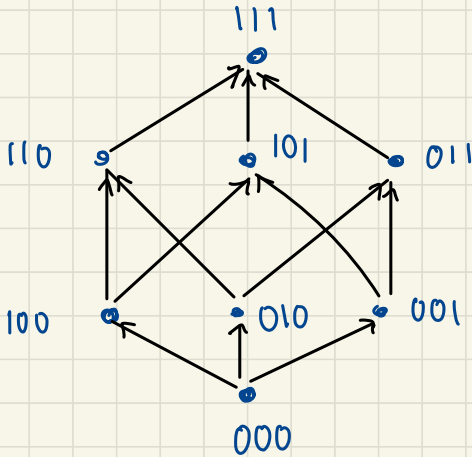
# nodes on level  $k$ :

$$\binom{n}{k}$$

edges:

$x \rightarrow y$

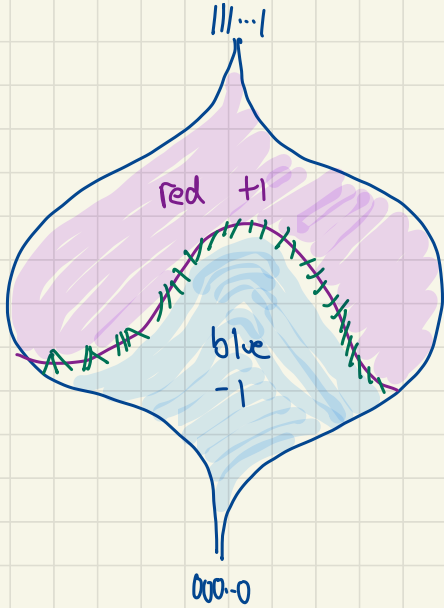
if flip one 0 in  $x$  to a 1  
to get  $y$



# nodes:  $2^n$   
# edges:  $\frac{n \cdot 2^n}{2}$

example for  $n=3$

# Monotone Functions on Boolean Cube



monotone  $\Rightarrow$  no blue above any red

$$X \leq Y \text{ if } \forall i \ x_i \leq y_i$$

$f$  monotone if

$$\forall X \leq Y, f(X) \leq f(Y)$$

Influence of  $f$ :

$$Inf_i(f) = \frac{\# \text{ red-blue edges in } i^{\text{th}} \text{ dir}}{2^{n-1}}$$

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

$\leftarrow$   $x$  with  $i^{\text{th}}$  bit flipped

$$Inf(f) = \frac{\# \text{ red-blue edges}}{2^n}$$

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

Thm  $f$  monotone  $\Rightarrow \inf_i (f) = f(\{1\})$

Thm majority fctn  $f(x) \equiv \text{sign}(\sum_{i=1}^n x_i)$  (odd  $n$ )  
maximizes influence among  
monotone fctns

Pfs on h.w.