Deterministic approximate counting for degree-2 polynomial threshold functions

Simons Workshop on
Real Analysis in Testing, Learning and Inapproximability

Rocco A. Servedio
Columbia University

Joint work with

Anindya De
UC Berkeley

Ilias Diakonikolas
U Edinburgh

Approximate Counting

Much work on approximately counting combinatorial structures:

- Given an n-by-n bipartite graph G, how many perfect matchings?
- Given an n-node bounded-degree graph G, how many k-colorings?
- etc.

Also much work (including this work) on approximately counting **satisfying assignments** of Boolean functions:

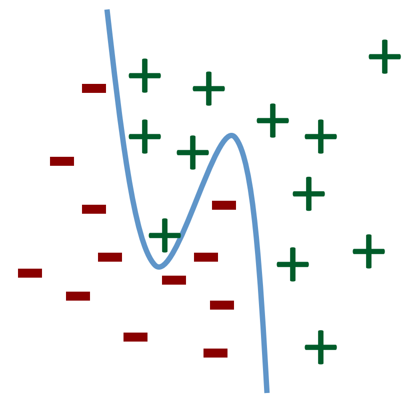
- Given a poly(n)-term DNF, how many satisfying assignments?
- Given an LTF, how many satisfying assignments?
- etc.

PTFs and LTFs

Degree-d **polynomial threshold function** (PTF): sign of a degree-d polynomial

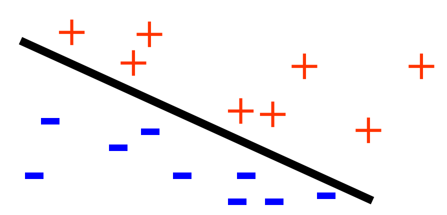
$$f : \{-1,1\}^n \rightarrow \{-1,1\}$$

$$f(x) = \mathrm{sign}(p(x_1, \ldots, x_n))$$



p a degree-d polynomial. Can assume it's multilinear.

Linear threshold function (LTF): degree d is 1.



Randomness

Very useful for approximately counting satisfying assignments!

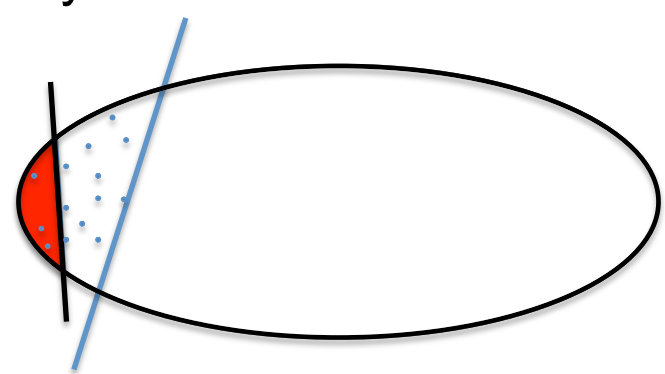
Example: LTFs

Input: an LTF $f(x) = sign(\sum_{i=1}^{n} w_i x_i - \theta)$

Output: a value $\widehat{p}$ such that $\widehat{p} \in [(1-\varepsilon)p, (1+\varepsilon)p]$ where

$$p = \Pr_{x \in \{-1,1\}^n}[f(x) = 1]$$

- [MorrisSinclair99]: sophisticated MCMC analysis
- [Dyer03]: elementary randomized algorithm & analysis using "dart throwing" & dynamic programming



Both approaches give $\mathrm{poly}(n, 1/\varepsilon)$-time algorithms.

A glorious success story: **deterministic** approximate counting for LTFs

More recently, $\mathrm{poly}(n, 1/\varepsilon)$-time **deterministic** (!) algorithms have been obtained for LTFs.

- [GopalanKlivansMeka10] : clever approximation of LTFs by read-once branching programs
- [StefankovicVempalaVigoda10]: clever use of dynamic programming

This work:
Approximately counting satisfying assignments for **degree-2 PTFs**

Input: a degree-2 PTF $f(x) = \mathrm{sign}(q(x))$
Output: a good approximation of $p = \Pr_{x \in \{-1,1\}^n}[f(x) = 1]$

Note: efficient multiplicative $(1 \pm \varepsilon)$-approximation of p is probably impossible, even using randomness…

…if you can distinguish $p = 0$ from $p > 0$, you can solve MAX-CUT: given $G = (V, E)$, the degree-2 polynomial

$$q(x) = (|E| - \sum_{\{i,j\} \in E} x_i x_j)/2 - k$$

is nonnegative iff $x \in \{-1,1\}^n$ specifies a cut of size at least k.

Additive approximation

Input: a degree-2 PTF $f(x) = \mathrm{sign}(q(x))$
Output: a good approximation of $p = \Pr_{x \in \{-1,1\}^n}[f(x) = 1]$

So, let's lower our standards: only seek an **additive** approximation $\widehat{p}$ such that $|p - \widehat{p}| \leq \varepsilon$

Good news: trivial randomized algorithm (sample assignments uniformly) works in $\mathrm{poly}(n, 1/\varepsilon)$ time!

Not so good news: this algorithm really, really uses randomness – and has nothing to do with degree-2 PTFs.

Motivation

Feels like the “right” problem for degree-2 PTF satisfying assignments (multiplicative approximation too hard; randomized additive approximation too easy)

Solving this problem for degree-2 PTFs forces us to understand them somehow

We like derandomizing things (and we like understanding degree-2 PTFs)

Main results of this work

Theorem: There is a $\mathrm{poly}(n, 2^{\mathrm{poly}(1/\varepsilon)})$-time deterministic algorithm which, on input any degree-2 PTF $f(x) = \mathrm{sign}(q(x))$ over $\{-1,1\}^n$, outputs a value $\widehat{p}$ such that $|p - \widehat{p}| \leq \varepsilon$, where $p = \Pr[f(x) = 1]$.

For "regular" degree-2 PTFs (each individual variable's influence on q is a small fraction of the total), the algorithm is an FPTAS:

Theorem: If q is an ε^9-regular polynomial, the algorithm runs in time $\mathrm{poly}(n, 1/\varepsilon)$.

Previous work on these types of questions

$d = 1$ case (LTFs): discussed already.

Can also do deterministic approximate counting using **unconditional PRGs** for degree-2 (or degree-d) PTFs.

PRG of size S for class $\mathcal{C}$ of functions $f : \{-1,1\}^n \to \{-1,1\}$: explicit set of points $X \subset \{-1,1\}^n, \ |X| = S$ such that

$$\left| \Pr_{x \in X}[f(x) = 1] - \Pr_{x \in \{-1,1\}^n}[f(x) = 1] \right| \leq \varepsilon \text{ for all } f \text{ in } \mathcal{C}.$$

Given such a PRG, can deterministically approximately count satisfying assignments of functions in $\mathcal{C}$ in time $\mathrm{poly}(n, S)$.

Unconditional PRGs for LTFs, PTFs

Much recent work on these:

- [DiakonikolasGopalanJaiswalSViola09]: size $n^{\tilde{O}(1/\varepsilon^2)}$ for LTFs (bounded independence)
- [DiakonikolasKaneNelson10]: size $n^{\mathrm{poly}(1/\varepsilon)}$ for degree-2 PTFs (bounded independence)
- [MekaZuckerman10]: size $\mathrm{poly}(n) \cdot \mathrm{quasipoly}(1/\varepsilon)$ for LTFs, size $n^{1/\varepsilon^{O(d)}}$ for degree-d PTFs
- [Kane12]: size $n^{\mathrm{poly}(1/\varepsilon)}$ for degree-d PTFs

For degree-2 PTFs, none of these PRGs give fixed $\mathrm{poly}(n)$-time approximate counting algorithms. (Equivalently, none work for $\varepsilon = o_n(1)$ in $\mathrm{poly}(n)$ time.)

PRGs are a "one hand tied behind the back" approach to deterministic approximate counting – they don't even look at the input!

Talk overview

✓ Introduction, motivation, statement of result

Rest of talk: proof of main result.

- From Gaussian to Boolean: suffices to solve Gaussian problem
- Solving the Gaussian problem:
  - transforming input polynomial to a "nice" form
  - counting Gaussian satisfying assignments for "nice" polynomials

The Gaussian problem

Recall main result -- counting **Boolean** satisfying assignments:

Theorem: There is a $\mathrm{poly}(n, 2^{\mathrm{poly}(1/\varepsilon)})$-time deterministic algorithm which, on input any degree-2 PTF $f(x) = \mathrm{sign}(q(x))$ over $\{-1,1\}^n$, outputs a value $\widehat{p}$ such that $|p - \widehat{p}| \leq \varepsilon$, where $p = \Pr_{x \in \{-1,1\}^n}[f(x) = 1]$

Key intermediate result -- counting **Gaussian** sat assignments:

Theorem: There is a $\mathrm{poly}(n, 1/\varepsilon)$ -time deterministic algorithm which, on input any degree-2 PTF $f(x) = \mathrm{sign}(q(x))$ over $\mathbf{R}^n$, outputs a value $\widehat{p}$ such that $|p - \widehat{p}| \leq \varepsilon$, where $p = \Pr_{x \sim N(0,1)^n}[f(x) = 1]$

From Gaussian to Boolean

Once we have the Gaussian counting result,

- can use “invariance principle” [MosselO’ DonnellOleszkiewicz05] to get $\mathrm{poly}(n, 1/\varepsilon)$ -time algorithm for “regular” degree-2 polynomials over Boolean cube;

- can use “PTF regularity lemma” [DiakonikolasSTanWan10, HarshaKlivansMeka09] to decompose any degree-2 PTF over the cube into $\exp(\mathrm{poly}(1/\varepsilon))$ many degree-2 PTFs almost all of which are $\mathrm{poly}(\varepsilon)$-regular or close to constant.

Follows (what is getting to be a) well-worn path for LTF, PTF problems.

Road map

- ✓ Introduction, motivation, statement of result, application to deterministically approximating moments
- ✓ From Gaussian to Boolean: suffices to solve Gaussian problem
- Solving the Gaussian problem:
  - transforming input polynomial to an equivalent polynomial which has a "nice" (decoupled junta) form
  - counting Gaussian satisfying assignments for "nice" polynomials

Constructing a equivalent "decoupled junta" degree-2 PTF

Theorem: There is a $\mathrm{poly}(n, 1/\varepsilon)$-time deterministic algorithm *Construct-Gaussian-Junta* which, given any degree-2 polynomial $q(x)$, outputs a degree-2 polynomial

$$\tilde{q}(x) = \sum_{i=1}^{K} (\lambda_i x_i^2 + \mu_i x_i) + C,$$

where $K = \tilde{O}(1/\varepsilon^4)$, such that

$$\left| \Pr_{x \sim N(0,1)^n}[q(x) \geq 0] - \Pr_{y \in N(0,1)^K}[\tilde{q}(y) \geq 0] \right| \leq \varepsilon.$$

High-level proof strategy: "critical index"-type analysis (reminiscent of "regularity lemma for LTFs" that's implicit in [S07]) with a few twists.

High-level sketch of "critical index analysis for LTFs"

Consider a halfspace over $\{-1,1\}^n$,

$$\mathrm{sign}(w \cdot x - \theta), \quad w_1 \geq \cdots \geq w_n \geq 0.$$

1. If w is regular (w_1 small compared to $\|w\|_2$) then for $x \sim \{-1,1\}^n$, $w \cdot x$ is distributed like a Gaussian ☺

2. If w not regular (w_1 large compared to $\|w\|_2$), "set w_1 aside" and consider $(w_2, \ldots, w_n)$: the 2-norm decreased by a lot. Repeat.

If have K = "many" iterations of step 2, remaining 2-norm of $(w_K, \ldots, w_n)$ is negligible ☺

We do something similar in our $N(0,1)^n$, degree-2 PTF setting.

Useful tool: Chatterjee's CLT

For $q(x) = x^T Ax + b \cdot x + c$ a degree-2 polynomial, write $\lambda_{\max}(q)$ to denote the largest-magnitude eigenvalue of A.

Theorem: Let $q(x)$ be a degree-2 PTF over $x \sim N(0,1)^n$. If $|\lambda_{\max}(q)| \leq \varepsilon\sqrt{\mathrm{Var}[q]}$, then distribution of $q(x)$ is $O(\varepsilon)$-close to the Gaussian distribution $N(\mathbf{E}[q], \mathrm{Var}[q])$ in total variation distance, hence

$$\left| \Pr_{x\sim N(0,1)^n}[q(x) \geq 0] - \Pr_{y\sim N(0,1)}[\sqrt{\mathrm{Var}[q]}y + \mathbf{E}[q] \geq 0] \right| \leq O(\varepsilon)$$

Follows from recent CLT of [Chatterjee09] (proved via Stein's method)

"$|\lambda_{\max}(q)| \leq \varepsilon\sqrt{\mathrm{Var}[q]}$" condition: analogue of having vector w be ε-regular in the $\{-1,1\}^n$ LTF setting.

Proof sketch

Want to prove:

Theorem: There is a $\mathrm{poly}(n, 1/\varepsilon)$ -time deterministic algorithm *Construct-Gaussian-Junta* which, given any degree-2 polynomial $q(x)$, outputs a degree-2 polynomial $\tilde{q}(x) = \sum_{i=1}^{K} (\lambda_i x_i^2 + \mu_i x_i) + C$, where $K = \tilde{O}(1/\varepsilon^4)$, such that

$$\left| \Pr_{x \sim N(0,1)^n}[q(x) \geq 0] - \Pr_{y \in N(0,1)^K}[\tilde{q}(y) \geq 0] \right| \leq \varepsilon.$$

Algorithm starts by (approximately) computing largest eigenvalue/ eigenvector pair $\lambda_1, v^{(1)}$.

If $|\lambda_1| \leq \varepsilon\sqrt{\mathrm{Var}[q]}$, can achieve $K = 1$: output poly $\sqrt{\mathrm{Var}[q]}y_1 + \mathbf{E}[q]$

Typical case is that $|\lambda_1| > \varepsilon\sqrt{\mathrm{Var}[q]}$ (corresponds to having $|w_1| > \varepsilon\|w\|_2$ in the $\{-1,1\}^n$ LTF setting.)

Proof sketch, cont.

Theorem: There is a $\mathrm{poly}(n, 1/\varepsilon)$-time deterministic algorithm *Construct-Gaussian-Junta* which, given any degree-2 polynomial $q(x)$, outputs a degree-2 polynomial $\tilde{q}(x) = \sum_{i=1}^{K}(\lambda_i x_i^2 + \mu_i x_i) + C$, where $K = \tilde{O}(1/\varepsilon^4)$, such that $\left| \Pr_{x \sim N(0,1)^n}[q(x) \geq 0] - \Pr_{y \in N(0,1)^K}[\tilde{q}(y) \geq 0] \right| \leq \varepsilon.$

If $|\lambda_1| > \varepsilon\sqrt{\mathrm{Var}[q]}$: Define new $N(0,1)$ variable $y_1 = v^{(1)} \cdot x$.
Rewrite $q(x)$ as $(n+1)$-variable polynomial

$$\overbrace{\underbrace{\lambda_1 y_1^2 + \mu_1 y_1}_{\text{Corresponds to “setting } w_1 \text{ aside” in the } \{-1,1\}^n \text{ LTF setting}} + r_1(y_1, x_1, \ldots, x_n)}^{\text{distributed identically to } q(x) \text{ for } (y_1, x_1, \ldots, x_n) \sim N(0,1)^{n+1}}$$

Can show

- r_1 (essentially) does not depend on y_1
- $\mathrm{Var}[r_1] \leq (1-\varepsilon^4)\mathrm{Var}[q]$ (corresponds to 2-norm of $(w_2, \ldots, w_n)$ being "a lot" smaller than $\|w\|_2$ in the $\{-1,1\}^n$ LTF setting)

Remove from r_1 all terms that contain y_1 , and repeat on r_1 .

2nd stage:

- If $|\lambda_{\max}(r_1)| \leq \varepsilon\sqrt{\mathrm{Var}[r_1]}$, stop and output the polynomial

$$\lambda_1 y_1^2 + \mu_1 y_1 + \sqrt{\mathrm{Var}[r_1]}y_2 + \mathbf{E}[r_1]$$

- If $|\lambda_{\max}(r_1)| > \varepsilon\sqrt{\mathrm{Var}[r_1]}$, continue building the decoupled polynomial:

$$\lambda_1 y_1^2 + \mu_1 y_1 + \lambda_2 y_2^2 + \mu_2 y_2 + r_2(y_2, x_1, \ldots, x_n)$$

As before, r_2 essentially does not depend on y_2, and variance again goes down by $(1-\varepsilon^4)$ factor. Continue to 3rd stage.
Etc.

Proof sketch, concluded

If loop exits at some stage $K' \leq K \stackrel{\text{def}}{=} \tilde{O}(1/\varepsilon^4)$, done.

Otherwise, have

$$\sum_{i=1}^{K} \lambda_i y_i^2 + \mu_i y_i + r_K(x_1, \ldots, x_n)$$

where $\mathrm{Var}[r_k] \leq \varepsilon$. Can ignore r_K and incur error at most ε.

Concludes sketch of theorem:

Theorem: There is a $\mathrm{poly}(n, 1/\varepsilon)$ -time deterministic algorithm *Construct-Gaussian-Junta* which, given any degree-2 polynomial $q(x)$, outputs a degree-2 polynomial $\tilde{q}(x) = \sum_{i=1}^{K}(\lambda_i x_i^2 + \mu_i x_i) + C$, where $K = \tilde{O}(1/\varepsilon^4)$, such that

$$\left| \Pr_{x \sim N(0,1)^n}[q(x) \geq 0] - \Pr_{y \in N(0,1)^K}[\tilde{q}(y) \geq 0] \right| \leq \varepsilon.$$

Almost done…

- ✓ Introduction, motivation, statement of result, application to deterministically approximating moments
- ✓ From Gaussian to Boolean: suffices to solve Gaussian problem
- Solving the Gaussian problem:
  - ✓ transforming input polynomial to an equivalent "nice" (decoupled junta) form
  - counting Gaussian satisfying assignments for decoupled junta polynomials

Counting Gaussian junta satisfying assignments

Given a Gaussian junta, can count efficiently:

Theorem: There is a $\mathrm{poly}(1/\varepsilon)$-time deterministic algorithm which, on input any degree-2 junta PTF $q(y) = \sum_{i=1}^{K}(\lambda_i y_i^2 + \mu_i y_i) + C$, with $K = \tilde{O}(1/\varepsilon^4)$, outputs a value $\widehat{p}$ such that

$$\left| \widehat{p} - \Pr_{y \sim N(0,1)^K}[\mathrm{sign}(q(y)) = 1] \right| \leq \varepsilon.$$

Probably many ways to do this. An elementary approach: discretize Gaussians, discretize polynomial, use dynamic programming

Counting Gaussian junta satisfying assignments

Theorem: There is a $\mathrm{poly}(1/\varepsilon)$ -time deterministic algorithm which, on input any degree-2 junta PTF $q(y) = \sum_{i=1}^{K}(\lambda_i y_i^2 + \mu_i y_i) + C$, with $K = \tilde{O}(1/\varepsilon^4)$, outputs a value $\widehat{p}$ such that $\left|\widehat{p} - \Pr_{y \sim N(0,1)^K}[\mathrm{sign}(q(y)) = 1]\right| \leq \varepsilon.$

1. Round coefficients of q to integer multiples of $\mathrm{poly}(\varepsilon)$; call resulting poly $\tilde{q}$
   - Using Gaussian concentration and anti-concentration, can show this changes probability by at most $O(\varepsilon)$

2. Discretize each Gaussian random variable:
   $y_i \sim N(0,1)$ → $\tilde{y}_i$ uniform over $\{t_1, \ldots, t_M\}, M = \mathrm{poly}(1/\varepsilon)$
   - Changes probability by at most $O(\varepsilon)$

3. Using dynamic programming, can exactly compute $\Pr[\tilde{q}(\tilde{y}) \geq 0]$ in $\mathrm{poly}(1/\varepsilon)$ time.

Summary

Gave a deterministic EPTAS for counting degree-2 PTF satisfying assignments: $\mathrm{poly}(n, 2^{\mathrm{poly}(1/\varepsilon)})$ time. Fully polynomial for Gaussian inputs, also for regular PTFs over Boolean inputs.

After lunch Anindya will speak about recent follow-up work: an efficient deterministic algorithm for counting satisfying assignments of

$$f(x) = J(\mathrm{sign}(q_1(x)), \ldots, \mathrm{sign}(q_k(x)))$$

for any $J : \{-1,1\}^k \to \{-1,1\}$, any $k = O(1)$.

Thank you