**CS 880: Pseudorandomness and Derandomization** 2/25-27/2013

# Lecture 11: Applications of Expanders

Instructors: Holger Dell and Dieter van Melkebeek Scribe: Mahnaz Akbari and Nick Pappas

In this lecture we discuss a central application of expanders in the context of derandomization, namely randomness-efficient *confidence boosting*, i.e., reducing the error probability of a randomized algorithm without increasing the need for random bits by too much. We present two approaches – a deterministic one and a randomized one. Deterministic confidence boosting requires no additional randomness but takes more time to achieve a certain level of confidence. Randomized confidence boosting takes less time but requires a small amount of additional randomness.

Recall the setting of confidence boosting from Lecture 5. Given a randomized algorithm $A(x, \rho)$ with error at most $\frac{1}{2} - \eta$, we want to construct a randomized algorithm $A'(x, \rho')$ for the same problem but with error at most $\delta$ such that (i) the number of random bits $r' \doteq |\rho'|$ that $A'$ needs is not much larger than the $r \doteq |\rho|$ random bits $A$ needs, and (ii) the running time of $A'$ is not much larger than the running time of $A$. The new algorithm $A'(x, \rho')$ runs $A(x, \rho_i)$ for $t$ random bit sequences $\rho_i$, $i \in [t]$, and combines the results in an appropriate way. If the underlying problem has a unique solution, the combining function is simply the plurality vote.

In Lecture 5 we analyzed confidence boosting when the $\rho_i$ are chosen (a) independently uniformly and (b) pairwise uniformly. In this lecture we do the same for two ways of obtaining the $\rho_i$ as vertices of an explicit expander $G$ of size $N = 2^r$, equating the vertices of $G$ with the bit strings in $\{0, 1\}^r$.

## 1 Deterministic Confidence Boosting

For deterministic confidence boosting we take the $\rho_i$ as the neighbors of $\rho'$ in $G$. We denote the set of neighbors of $\rho'$ as $\Gamma(\rho')$. Note that no additional random bits are required: We only need to pick $\rho'$ uniformly at random and $|\rho'| = \rho_i$, so we have $r' = r$. Intuitively, because $G$ is expanding the neighbors are spread out fairly well over the graph, which means we are close to the setting where we pick the $\rho_i$ independently uniformly. We analyze the resulting reduction in error using the quasi-randomness property of expanders.

**Lemma 1.** *Suppose that the randomized algorithm $A$ on input $x$ uses $r$ random bits and outputs $y$ with probability $\frac{1}{2} + \eta$. Let $G$ be a regular multigraph on $\{0, 1\}^r$. Then*

$$\Pr_{\rho' \leftarrow U_r}[\text{plurality}_{\rho \in \Gamma(\rho')} A(x, \rho) \neq y\}] \leq \lambda(G)^2 (\frac{1}{4} - \eta^2)/\eta^2.$$

*Proof.* Fix the input $x$ and the output $y$. Let $B$ denote the set of random bit strings that yield an output for $A$ other than $y$, and let $B'$ denote the corresponding set for the boosted algorithm, i.e.,

$$B = \{\rho : A(x, \rho) \neq y\} \text{ and } B' = \{\rho' : \text{plurality}_{\rho \in \Gamma(\rho')} A(x, \rho) \neq y\}.$$

We know that $\mu(B) = \frac{1}{2} - \eta$ and want to upper bound $\mu(B')$. Note that for every $\rho' \in B'$, $A(x, \rho)$ differs from $y$ for at least half of the neighbors $\rho \in \Gamma(\rho')$, i.e., $|\Gamma(\rho') \cap B| \geq \frac{d}{2}$, where $d$ denotes the degree of $G$. Therefore,

$$|E(B, B')| \geq |B'| \cdot \frac{d}{2}. \tag{1}$$

The quasi-randomness property (Theorem 4 from Lecture 9) of $G$ with $S = B$ and $T = B'$ tells us that

$$\left| \frac{|E(B,B')|}{dN} - \mu(B)\mu(B') \right| \leq \lambda(G) \cdot \sqrt{\mu(B)(1-\mu(B))} \cdot \sqrt{\mu(B')(1-\mu(B'))}. \tag{2}$$

By the above we know that $\frac{|E(B,B')|}{dN} \geq \frac{\mu(B')}{2}$ and that $\mu(B)\mu(B') \leq \frac{\mu(B')}{2}$, so we can drop the absolute value signs on the left-hand side of (2). Using (1) we obtain from (2) that

$$\begin{aligned} \mu(B') \cdot \eta &= \frac{\mu(B')}{2} - (\frac{1}{2} - \eta)\mu(B') \\ &\leq \frac{|E(B,B')|}{dN} - \mu(B)\mu(B') \\ &\leq \lambda(G) \cdot \sqrt{\mu(B)(1-\mu(B))} \cdot \sqrt{\mu(B')(1-\mu(B'))} \\ &= \lambda(G) \cdot \sqrt{\frac{1}{4} - \eta^2} \cdot \sqrt{\mu(B')(1-\mu(B'))}. \end{aligned}$$

After rearranging and simplifying this gives us

$$\sqrt{\mu(B')} \leq \frac{\lambda(G)}{\eta} \cdot \sqrt{\frac{1}{4} - \eta^2},$$

which is equivalent to the bound in the statement. □

Using the fully explicit expanders from Lecture 10, Lemma 1 leads to the following confidence boosting result. For simplicity we state it for decision procedures only.

**Theorem 2 (Deterministic Confidence Boosting).** *Given a randomized decision algorithm $A$ that uses $r$ random bits, has error at most $\frac{1}{2} - \eta$, and runs in time $t$, there exists a randomized decision algorithm $A'$ for the same problem that has error at most $\delta$, uses $r$ random bits, and runs in time* $\text{poly}(\frac{1}{\delta\eta}) \cdot (\text{poly}(r) + t)$.

*Proof.* We let $A'(x, \rho')$ output the majority vote of $A(x, \rho)$ over $\rho \in \Gamma(\rho')$. In order to guarantee that the error bound given by Lemma 1 is no more than $\delta$, it suffices that

$$\lambda(G) \leq \sqrt{\delta} \cdot \eta. \tag{3}$$

The fully explicit expanders $G$ given by Theorem 4 from Lecture 10 satisfy $\lambda(G) \leq \lambda$ for some constant $\lambda < 1$, and have constant degree $d$. By taking their $s$-th power, i.e., by considering all walks of length $s$ in $G$, we obtain graphs $G' = G^s$ with $\lambda(G') \leq \lambda^s$ and degree $d' = d^s$. Setting $s = \frac{\log(1/(\sqrt{\delta}\eta))}{\log(1/\lambda_0)}$ ensures (3) and yields degree $d' = \text{poly}(\frac{1}{\delta\eta})$. The resulting algorithm $A'(x, \rho')$ runs $A(x, \rho_i)$ on $d'$ random bit sequences $\rho_i$, and additionally needs to compute at most $d'$ times a neighbor of a vertex in $G$. Since $G$ is fully explicit and has $N = 2^r$ vertices, the latter can be done in time polynomial in $r$ per neighbor. Thus, the overall running time is

$$d' \cdot (\text{poly}(r) + t) = \text{poly}(\frac{1}{\delta\eta}) \cdot (\text{poly}(r) + t).$$

Strictly speaking, Theorem 4 from Lecture 10 only guarantees the existence of $G$ on $N = 2^r$ vertices for even $r$. However, for odd $r$ we can use $G$ with $N = 2^{r+1}$ vertices, and on a given $\rho' \in \{0,1\}^r$ use all of $\Gamma(\rho'0) \cup \Gamma(\rho'1)$ as the $\rho_i$. The overall majority vote can only be incorrect if the majority over at least one of $\Gamma(\rho'0)$ or $\Gamma(\rho'1)$ is incorrect. Thus, by a union bound, the error probability of this modified procedure is at most $2\delta$. □

For decision procedures with *one-sided error* there is a better combining function. If $A$ has no false positives, $A'$ can accept as soon as at least one of the runs of $A$ accepts; $A'$ rejects otherwise. Then $A'$ also has no false positives, and $A'$ errs iff $A$ errs on all runs. This allows us to handle error rates for $A$ that are close to 1. We leave the analysis as an exercise.

**Exercise 1.** *Show that if $A$ has one-sided error, then the upper bound of $\frac{1}{2} - \eta$ in the statement of Theorem 2 can be relaxed to $1 - \eta$ and $A'$ can be made one-sided, without any further changes to the statement. Hint: First show how to modify Lemma 1.*

# 2 Randomized Confidence Boosting

Instead of picking the $\rho_i$ as the neighbors of some vertex $\rho$ of $G$, we can pick them as the vertices on a random walk starting from $\rho$. Intuitively, this improves the confidence more because we get further away from the start vertex $\rho$, and therefore closer to the situation where we pick the $\rho_i$ independently uniformly. The number of random bits needed for the random walk of length $s-1$ (consisting of $s$ vertices including the start vertex $\rho$) is $r + (s-1) \cdot \log d$, where $d$ denotes the degree of $G$.

## 2.1 One-Sided Error

We first analyze this boosting procedure in the case of decision algorithms with one-sided error. The following analysis uses the matrix decomposition given by Proposition 3 from Lecture 9:

$$M_G = (1 - \lambda(G)) \cdot M_K + \lambda(G) \cdot \Delta, \tag{4}$$

where $K$ denotes the complete digraph with self-loops, and $\|\Delta\|_2 \leq 1$.

**Lemma 3.** *Suppose that the randomized algorithm $A$ on input $x$ uses $r$ random bits and outputs $y$ with probability $\eta$. Let $G$ be a regular multigraph on $\{0,1\}^r$, and consider a random walk of length $s-1$ in $G$. Then*

$$\Pr_{\rho' \leftarrow U_r}[(\forall k \in [s]) A(x, \rho_k) \neq y] \leq (1-\eta)\left(1 - (1-\lambda(G))(1-\sqrt{1-\eta})\right)^{s-1}, \tag{5}$$

*where $\rho'$ describes a random walk of length $s-1$, and $\rho_k$ denotes the $k$-th vertex on the random walk.*

*Proof.* Fix the input $x$ and the output $y$. Let $B$ denote the set of random bit strings that yield an output for $A$ other than $y$. Let $B' = \{\rho' : (\forall k \in [s])\, \rho_k \in B\}$. We have that $\mu(B) = 1 - \eta$ and want to upper bound $\mu(B')$.

We first express $\mu(B')$ in linear-algebraic terms using $M_G$. Let $N \doteq 2^r$, and let $P \in \{0,1\}^{N \times N}$ denote the projection onto $B$:

$$P_{ij} = \begin{cases} 1 & \text{if } i = j \in B \\ 0 & \text{otherwise.} \end{cases}$$

The key is the following claim.

*Claim.* For every $i, j \in [N]$ and every integer $\ell \geq 0$, $((PM_G)^\ell P)_{ij}$ equals the probability that a random walk of length $\ell$ in $G$ starting from $j$ stays entirely within $B$ and ends in $i$.

Intuitively, the effect of each application of $P$ is to annihilate all walks that fall outside of $B$ at that point in time, and to leave all others unaffected. The formal proof follows by induction on $\ell$ and is left as an exercise.

By Claim 1 we can write the probability that the random walk stays entirely within $B$ and ends in $i$ as $((PM_G)^{s-1}PU_N)_i$. Therefore,

$$\begin{aligned} \mu(B') &\doteq \Pr[(\forall k \in [s])\, \rho_k \in B] \\ &= \|(PM_G)^{s-1}PU_N\|_1 \\ &\leq \sqrt{N} \cdot \|(PM_G)^{s-1}PU_N\|_2 \\ &\leq \sqrt{N} \cdot \|PM_G\|_2^{s-1} \cdot \|PU_N\|_2, \end{aligned} \tag{6}$$

where we used properties of the 1-norm and the 2-norm.

By (4) and the triangle inequality

$$\|PM_G\|_2 \leq (1-\lambda(G))\|PM_K\|_2 + \lambda(G)\|P\Delta\|_2. \tag{7}$$

As for the first term in (7), consider an arbitrary column vector $v \in \mathbb{R}^N$. The vector $M_K v$ is parallel to $U_N$; more precisely, $M_K v = (\sum_{i=1}^N v_i)U_N$. It follows that

$$\|PM_K v\|_2 = \|P(\sum_{i=1}^N v_i)U_N\|_2 \leq \|v\|_1 \cdot \|PU_N\|_2 \leq \sqrt{N}\|v\|_2\|PU_N\|_2.$$

Since $PU_N = \frac{1}{N}I_B$,

$$\|PU_N\|_2 = \frac{\sqrt{|B|}}{N}. \tag{8}$$

Thus,

$$\|PM_K v\|_2 \leq \sqrt{N} \cdot \|v\|_2 \cdot \frac{\sqrt{|B|}}{N} = \sqrt{\mu(B)} \cdot \|v\|_2.$$

Since $v$ is arbitrary this means that

$$\|PM_K\|_2 \leq \sqrt{\mu(B)}.$$

As for the second term in (7), we have that $\|P\Delta\|_2 \leq \|P\|_2 \cdot \|\Delta\|_2 \leq 1$. Plugging in the upper bounds for both terms on the right-hand side of (7) yields

$$\|PM_G\|_2 \leq (1-\lambda(G))\sqrt{\mu(B)} + \lambda(G) = 1 - (1-\lambda(G))(1-\sqrt{\mu(B)}). \tag{9}$$

Using this upper bound and (8) in (6), we conclude that

$$\mu(B') \leq \sqrt{N} \cdot \Big(1 - (1-\lambda(G))(1-\sqrt{\mu(B)})\Big)^{s-1} \cdot \frac{\sqrt{|B|}}{N},$$

which is equivalent to the bound claimed in the statement. □

The bound given by Lemma 3 can be further improved, namely as follows.

**Exercise 2.** *Show that $\mu(B')$ in the proof of Lemma 3 can be written as $\mu(B') = \|(PM_GP)^{s-1}PU_N\|_1$ and that $\|PM_GP\|_2 \leq 1-(1-\lambda(G))(1-\mu(B))$. Conclude that the right-hand side of (5) can be tightened to*

$$(1-\eta)\big(1-(1-\lambda(G))\eta\big)^{s-1}.$$

We can further simplify the latter upper bound using $1-x \leq \exp(-x)$ as follows:

$$(1-\eta)\big(1-(1-\lambda(G))\eta\big)^{s-1} \leq \big(1-(1-\lambda(G))\eta\big)^{s} \leq \exp\big(-(1-\lambda(G))\eta s\big).$$

Thus, it suffices to pick $s = \frac{\ln(1/\delta)}{(1-\lambda(G))\eta}$ in order to guarantee that the error of $A'$ is no more than $\delta$. If the expander family has constant degree and a spectral gap that is at least some positive constant, it follows that $s = O(\frac{\log(1/\delta)}{\eta})$, and the number of random bits needed is $r + O(s) = r + O(\frac{\log(1/\delta)}{\eta})$.

We can further reduce the number of random bits and remove the dependency on $\eta$ by first using deterministic confidence boosting up to a constant level, and then applying the random walk approach. This leads to the following confidence boosting result for decision procedures with one-sided error.

**Theorem 4 (Expander-Walk Confidence Boosting for One-Sided Error).** *Given a randomized decision algorithm $A$ that uses $r$ random bits, has one-sided error at most $1-\eta$, and runs in time $t$, there exists a randomized decision algorithm $A'$ for the same problem that has one-sided error at most $\delta$, uses $r + O(\log(\frac{1}{\delta}))$ random bits, and runs in time $\text{poly}(\frac{1}{\eta}) \cdot \log(\frac{1}{\delta}) \cdot (\text{poly}(r) + t)$.*

*Proof.* We first use Exercise 1 to reduce the error rate from $1-\eta$ down to $\frac{1}{2}$ while maintaining one-sidedness. Let $\tilde{A}$ denote the resulting decision algorithm. The amount of random bits of $\tilde{A}$ remains at $r$, and the running time of $\tilde{A}$ is $\text{poly}(\frac{1}{\eta}) \cdot (\text{poly}(r) + t)$.

Next we apply Lemma 3 to $\tilde{A}$ so as to further reduce the error from $\frac{1}{2}$ down to $\delta$. We use the fully explicit family of expanders $G$ given by Theorem 4 from Lecture 10, which has constant degree $d$ and $\lambda(G) \leq \lambda$ for some constant $\lambda < 1$. Using the improved analysis from Exercise 2, the overall error is at most $\delta$ for $s = 2\ln(\frac{1}{\delta})/(1-\lambda(G))$, which is $O(\log(\frac{1}{\delta}))$. The resulting algorithm $A'$ needs $r + (s-1)\log(d)$ random bits, which is $r + O(\log(\frac{1}{\delta}))$. In order to run $A'(x, \rho')$ we need to compute $s-1$ times a neighbor of a vertex in the expander $G$ with $N = 2^r$ vertices, and to run $\tilde{A}(x, \rho)$ for $s$ choices of $\rho$. The overall running time of $A'$ is

$$s \cdot \left(\text{poly}(r) + \text{poly}(\frac{1}{\eta}) \cdot (\text{poly}(r) + t)\right) = \text{poly}(\frac{1}{\eta}) \cdot \log(\frac{1}{\delta}) \cdot (\text{poly}(r) + t).$$

The issue that the family given by Theorem 4 from Lecture 10 only contains graphs $G$ with $N = 2^r$ vertices for even $r$ can be handled in a similar way as in the proof of Theorem 2. □

Compared to the deterministic boosting approach from Exercise 1, the dependency of the running time on $\frac{1}{\delta}$ is only logarithmic rather than polynomial. Figure 1 compares the various *randomized* approaches we have seen for reducing the one-sided error of an algorithm $A$ from $1-\eta$ down to $\delta$ in terms of (a) the number of truly random bits needed and (b) the number of times $A$ is run. Lines (i) and (ii) in Figure 1 were analyzed in Lectures 3 and 5 for two-sided error. Lines (iii) and (iv) were analyzed earlier in this lecture. We leave it as an exercise to prove that (v) can be realized; we refer to [Gol11, Appendix C] for matching lower bounds.

**Exercise 3.** *Show how to realize (v) by following the two-phased approach from Theorem 4 but using Ramanujan graphs for the deterministic phase.*

|  | method | number of random bits | number of runs |
|---|---|---|---|
| (i) | fully independent runs | $O(r \cdot \log(\frac{1}{\delta}) \cdot \frac{1}{\eta})$ | $O(\log(\frac{1}{\delta}) \cdot \frac{1}{\eta})$ |
| (ii) | pairwise independent runs | $O(r + \log(\frac{1}{\delta}) + \log(\frac{1}{\eta}))$ | $O(\frac{1}{\delta\eta})$ |
| (iii) | expander walk | $r + O(\log(\frac{1}{\delta}) \cdot \frac{1}{\eta})$ | $O(\frac{1}{\eta}\log(\frac{1}{\delta}))$ |
| (iv) | Theorem 4 | $r + O(\log(\frac{1}{\delta}))$ | $\mathrm{poly}(\frac{1}{\eta}) \cdot \log(\frac{1}{\delta})$ |
| (v) | optimal | $r + \Theta(\log(\frac{1}{\delta}))$ | $\Theta(\frac{1}{\eta}\log(\frac{1}{\delta}))$ |

Figure 1: Overview of randomized approaches for one-sided confidence boosting.

## 2.2 Two-Sided Error

For randomized decision algorithms with *two-sided error* we can combine the $s$ runs of the random walk by taking the majority vote, and for more general randomized algorithms computing a function by taking the plurality vote. The error analysis given in Lemma 3 extends as follows.

**Lemma 5.** *Suppose that the randomized algorithm $A$ on input $x$ uses $r$ random bits and outputs $y$ with probability $\frac{1}{2} + \eta$. Let $G$ be a regular multigraph on $\{0,1\}^r$, and consider a random walk of length $s-1$ in $G$ starting from $\rho'$. Then*

$$\Pr_{\rho' \leftarrow U_r}[\mathrm{plurality}_{k \in [s]} A(x, \rho_k) \neq y] \leq 2^s \left( (1 - \lambda(G)) \sqrt{\frac{1}{2} - \eta} + \lambda(G) \right)^{\frac{s}{2} - 1},$$

*where $\rho'$ describes a random walk of length $s-1$, and $\rho_k$ denotes the $k$-th vertex on the random walk.*

*Proof.* We follow the same outline as in the proof of Lemma 3 and borrow the notation $B$ for the bad set of $A$ on input $x$, and $P$ for the projection onto $B$. We have that $\mu(B) = \frac{1}{2} - \eta$ and want to upper bound $\mu(B')$ where $B' \doteq \{\rho' : \mathrm{plurality}_{k \in [s]} A(x, \rho_k) \neq y\}$.

Similar to Claim 1, for every $i, j \in [N]$, every integer $\ell \geq 0$, and every subset $L \subseteq [\ell]$, we can write the probability that a random walk of length $\ell$ in $G$ starting from $j$ ends in $i$ and has $\rho_k \in B$ for every $k \in L$ as

$$(Q_\ell M_G Q_{\ell-1} M_G \cdots M_G Q_2 M_G Q_1)_{ij},$$

where

$$Q_k = \begin{cases} P & \text{if } k \in B \\ I & \text{otherwise.} \end{cases}$$

It follows that

$$\begin{aligned}
\mu(B') &\leq \Pr[\rho_k \in B \text{ for at least } \frac{s}{2} \text{ of } k \in [s]] \\
&\leq \sum_{S \subseteq [s], |S| \geq s/2} \Pr[(\forall k \in S) \rho_k \in B] \\
&= \sum_{S \subseteq [s], |S| \geq s/2} \|(Q_\ell M_G Q_{\ell-1} M_G \cdots M_G Q_2 M_G Q_1) U_N\|_1 \\
&\leq \sum_{S \subseteq [s], |S| \geq s/2} \sqrt{N} \cdot ( \prod_{1<k\leq s} \|Q_k M_G\|_2) \cdot \|Q_1 U_N\|_2 \\
&\leq \sum_{S \subseteq [s], |S| \geq s/2} \sqrt{N} \|PM_G\|_2^{|S|-1} \cdot \|U_N\|_2 \text{ (since } \|Q_k M_G\|_2 \leq 1 \text{ and } \|Q_1\|_2 \leq 1) \\
&\leq 2^s \|PM_G\|_2^{\frac{s}{2}-1}.
\end{aligned}$$

The bound given in the statement then follows by (9). □

Lemma 5 leads to the following confidence boosting result for decision procedures with two-sided error.

**Theorem 6 (Expander-Walk Confidence Boosting for Two-Sided Error).** *Given a randomized decision algorithm $A$ that uses $r$ random bits, has error at most $\frac{1}{2} - \eta$, and runs in time $t$, there exists a randomized decision algorithm $A'$ for the same problem that has error at most $\delta$, uses $r + O(\log(\frac{1}{\delta}))$ random bits, and runs in time* $\text{poly}(\frac{1}{\eta}) \cdot \log(\frac{1}{\delta}) \cdot (\text{poly}(r) + t)$.

Compared to the deterministic boosting approach from Theorem 2, the dependency of the running time on $\frac{1}{\delta}$ is only logarithmic rather than polynomial. The proof of Theorem 6 follows along the same lines as the one of Theorem 4, using Lemma 5 instead of Lemma 3. We leave the proof as an exercise.

To compare the various randomized approaches for reducing the two-sided error of an algorithm $A$ from $\frac{1}{2} - \eta$ down to $\delta$, Figure 2 tabulates the number of truly random bits needed and the number of times $A$ is run.

| | method | number of random bits | number of runs |
|---|---|---|---|
| (i) | fully independent runs | $O(r \cdot \log(\frac{1}{\delta}) \cdot \frac{1}{\eta^2})$ | $O(\log(\frac{1}{\delta}) \cdot \frac{1}{\eta^2})$ |
| (ii) | pairwise independent runs | $O(r + \log(\frac{1}{\delta}) + \log(\frac{1}{\eta}))$ | $O(\frac{1}{\delta\eta^2})$ |
| (iii) | Theorem 6 | $r + O(\log(\frac{1}{\delta}))$ | $\text{poly}(\frac{1}{\eta}) \cdot \log(\frac{1}{\delta})$ |
| (iv) | optimal | $r + \Theta(\log(\frac{1}{\delta}))$ | $\Theta(\frac{1}{\eta^2} \log(\frac{1}{\delta}))$ |

Figure 2: Overview of randomized approaches for two-sided confidence boosting.

Lines (i) and (ii) in Figure 2 were analyzed in Lectures 3 and 5. and line (iii) in this lecture. We leave it as an exercise to prove that (iv) can be realized; we refer to [CEG95] for matching lower bounds.

**Exercise 4.** *Show how to realize (v) by following the approach of Theorem 6 but using Ramanujan graphs for the deterministic phase.*

The essential difference between the tables in Figure for 1 for one-sided error and Figure 2 for two-sided error is the term $\frac{1}{\eta}$ vs $\frac{1}{\eta^2}$.

### 2.3 Chernoff Bound

Confidence boosting is closely related to Chernoff bounds, which are strong concentration results for sums of independent random variables. There are several variants. For background, we state and prove a simple variant in which the random variables are indicator variables with the same expectation, and the deviation from the mean is absolute. This version is attributed to Hoeffding.

**Theorem 7 (Hoeffind Bound).** *Let $X_1, X_2, \ldots, X_s$ denote independent indicator variables with the same expected value $\mu$. For every $\epsilon \in [0, 1]$*

$$\Pr\left[(\sum_{i=1}^{s} X_i) - \mu s \geq \epsilon s\right] \leq \exp\left(-\frac{\epsilon^2 s}{4}\right).$$

Theorem 7 implies that

$$\Pr\left[\left|(\sum_{i=1}^{s} X_i) - \mu s\right| \geq \epsilon s\right] \leq 2\exp\left(-\frac{\epsilon^2 s}{4}\right).$$

This concentration result is exponentially stronger than the one we obtained in Lecture 5 using Chebyshev's inequality, but Chebyshev only requires pairwise independence, whereas Hoeffding requires full independence. We proved Chebyshev by transforming each side of the inequality using the mapping $f(x) = x^2$ and then applying Markov. To prove Hoeffding, we follow the same outline but use a mapping $f$ that grows much more quickly and still interacts well with the expectation we need for applying Markov. Specifically, we use an exponential transformation of the form $f(x) = \exp(ax)$ for an appropriately chosen value of $a$.

*Proof.* Let $a$ be any positive real. Since $f(x) = \exp(ax)$ is an increasing function, we have that

$$\sum_{i=1}^{s} X_i \geq (\mu + \epsilon)s \Leftrightarrow \exp\left(a\sum_{i=1}^{s} X_i\right) \geq \exp(a(\mu + \epsilon)s).$$

Since $f(x)$ is nonnegative, the random variable $\exp\left(a\sum_{i=1}^{s} X_i\right)$ is nonnegative so we can apply Markov's inequality. We get that

$$\begin{aligned}
\Pr\left[(\sum_{i=1}^{s} X_i) - \mu s \geq \epsilon s\right] &= \Pr\left[\sum_{i=1}^{s} X_i \geq (\mu + \epsilon)s\right] \\
&= \Pr\left[\exp\left(a\sum_{i=1}^{s} X_i\right) \geq \exp(a(\mu + \epsilon)s)\right] \\
&\leq \frac{\mathbb{E}[\exp(a\sum_{i=1}^{s} X_i)]}{\exp(a(\mu + \epsilon)s)}. \qquad (10)
\end{aligned}$$

The numerator can be upper bounded as follows.

$$\begin{aligned}
\mathbb{E}[\exp(a\sum_{i=1}^{s}X_i)] &= \mathbb{E}[\prod_{i=1}^{s}\exp(aX_i)] \\
&= \prod_{i=1}^{s}\mathbb{E}(aX_i) \text{ (by full independence)} \\
&= \prod_{i=1}^{s}(\mu\cdot e^a+(1-\mu)) \text{ (expectation of an indicator variable)} \\
&= (1+(e^a-1)\mu)^s \text{ (by rewriting)} \\
&\leq \exp((e^a-1)\mu s) \text{ (using } 1+x\leq e^x \text{ for } x=(e^a-1)\mu) \qquad (11)
\end{aligned}$$

For $a\in[0,\frac{1}{2}]$, we can upper bound $e^a$ by $e^a\leq 1+a+a^2$. Plugging in this upper bound into (11) and the latter into (10), we have that

$$\Pr\left[(\sum_{i=1}^{s}X_i)-\mu s\geq\epsilon s\right]\leq\exp\left((a+a^2)\mu s-a(\mu+\epsilon)s\right)\leq\exp(a(a-\epsilon)s).$$

The latter expression is minimized for $a=\frac{\epsilon}{2}\in[0,\frac{1}{2}]$, which gives us the upper bound stated in the theorem. □

Line (i) in Figure 2 can be rederived from Theorem 7 by setting $\epsilon=\eta$.

Note the critical role independence plays in the proof of Theorem 7. Such independence does not hold when the $X_i$ are obtained from a random walk in a low-degree graph. However, if the underlying graph is an expander, we are close enough to independence that a similar result still holds. This leads to the so-called Chernoff bound for expanders. We state a somewhat more general version than Theorem 7, namely one where the expected value does not have to be the same for every step.

**Theorem 8 (Chernoff Bound for Expanders).** *There exists a constant $b>0$ such that the following holds for every regular graph $G=(V,E)$. For $i\in[s]$, let $B_i\subseteq V$ and $X_i\doteq I[\rho_i\in B_i]$, where $\rho_1,\rho_2,\ldots,\rho_s$ denotes a random walk of length $s-1$ on $G$. For every $\epsilon\in[0,1]$*

$$\Pr\left[(\sum_{i=1}^{s}X_i)-(\sum_{i=1}^{s}\mathbb{E}[X_i])\geq\epsilon s\right]\leq\exp\left(-b\cdot(1-\lambda(G))\cdot\epsilon^2 s\right).$$

We refer to [Gil98] for a proof, which consists of a combination of the ingredients of the proof of Theorem 7 and the linear-algebraic analysis from the previous section.

Note that, modulo the constant $b$, Theorem 8 generalizes Theorem 7 as for the complete graph $G$, $\lambda(G)=0$. Line (iii) in Figure 2 can be rederived using Theorem 8.

## References

[CEG95] R. Canetti, G. Even, and O. Goldreich. Lower bounds for sampling algorithms for estimating the average. *Information Processing Letters*, 53:17–25, 1995.

[Gil98] D. Gillman. A Chernoff bound for random walks on expander graphs. *SIAM Journal on Computing*, 27:1203–1220, 1998.

[Gol11] Oded Goldreich. A sample of samplers: A computational perspective on sampling. In *Studies in Complexity and Cryptography*, pages 302–332. 2011.