Electronic Colloquium on Computational Complexity, Report No. 41 (2017)

# Explicit, almost optimal, epsilon-balanced codes

Amnon Ta-Shma [*]
Department of Computer Science,
Tel-Aviv University, Tel-Aviv 69978, Israel.
E–mail: amnon@tau.ac.il

## Abstract

The question of finding an epsilon-biased set with close to optimal support size, or, equivalently, finding an explicit binary code with distance $\frac{1-\varepsilon}{2}$ and rate close to the Gilbert-Varshamov bound, attracted a lot of attention in recent decades. In this paper we solve the problem almost optimally and show an explicit $\varepsilon$-biased set over $k$ bits with support size $O(\frac{k}{\varepsilon^{2+o(1)}})$. This improves upon all previous explicit constructions which were in the order of $\frac{k^2}{\varepsilon^2}$, $\frac{k}{\varepsilon^3}$ or $\frac{k^{5/4}}{\varepsilon^{5/2}}$. The result is close to the Gilbert-Varshamov bound which is $O(\frac{k}{\varepsilon^2})$ and the lower bound which is $\Omega(\frac{k}{\varepsilon^2 \log \frac{1}{\varepsilon}})$.

The main technical tool we use is bias amplification with the $s$-wide replacement product. The sum of two independent samples from an $\varepsilon$-biased set is $\varepsilon^2$ biased. Rozenman and Wigderson showed how to amplify the bias more economically by choosing two samples with an expander. Based on that they suggested a recursive construction that achieves sample size $O(\frac{k}{\varepsilon^4})$. We show that amplification with a long random walk over the $s$-wide replacement product reduces the bias almost optimally.

## 1 Introduction

The Gilbert-Varshamov (GV) bound states that for every $\delta > 0$, there exists a family $\{C_n\}$ of codes such that $C_n$ has length $n$, relative distance $\delta$ and relative rate $1 - H(\delta) - o(1)$, where $H$ is Shannon's entropy function. Finding an *explicit* construction approaching the Gilbert-Varshamov bound is one of the most important open problems in coding theory. For codes with distance close to half, the Gilbert-Varshamov bound shows that random linear codes of length $n = O(k/\varepsilon^2)$ are w.h.p. $[n, k, \frac{1-\varepsilon}{2}n]_2$ codes. Finding an explicit construction attaining this bound is, again, a major open problem.

One popular approach trying to build good binary codes close to the GV bound, is by starting with a code over a large alphabet and concatenating it with a good binary code. Concatenating the Reed Solomon code (RS) with the Hadamard code gives one of the constructions in AGHP with support size about $O(\frac{n^2}{k^2})$ [3]. Concatenating RS with the Wozencraft ensemble gives the Justensen code [13] with constant relative distance and constant relative rate. Starting with an AG code and concatenating with Hadamard over a field of size $O(\frac{1}{\varepsilon^2})$ gives a construction with about $O(\frac{k}{\varepsilon^3})$ support size. Taking the Hermitian code below the genus over a smaller field and concatenating with Hadamard gives the construction in [7] with support size $O((\frac{k}{\varepsilon^2})^{5/4})$.

All of the above bounds fall short of achieving the GV bound which is $O(\frac{k}{\varepsilon^2})$. Indeed, it seems the concatenation method is inherently limited and is not able to reach the GV bound. The attainable parameters

[*]Supported by Israel Science Foundation grant no. 994/14

ISSN 1433-8092

when concatenating with RS as the outer code are captured by the Zyablov bound, which for distance $\frac{1-\varepsilon}{2}$ gives code length $O(\frac{k}{\varepsilon^3})$ (see [2, Section 1]). Similarly, concatenating with any high genus AG code cannot attain the GV bound for distance close to half [6, Section 4]. Thus, another approach is needed.

Indeed, Naor and Naor [15], and later Alon et al. [2], suggested to solve the problem using *amplification* and *pseudo-randomness*. The idea is to start with a binary error-correcting code that has moderate distance (say, some constant relative distance) and *amplify* it to a binary error correcting code with a higher distance (say, close to half). This was done by Naor and Naor [15], and later Alon et al. [2], using expanders, or more generally dispersers. This approach also gives binary error correcting codes of length $n$, dimension $k$ and distance $\frac{1-\varepsilon}{2}$ with $n = O(\frac{k}{\varepsilon^3})$. Nevertheless, Alon et al. show that for a certain non-binary field size their construction lies above the Zyablov bound.

We now digress to give an equivalent representation of the problem in which it is easier to see what is amplified in the construction. We define two related problems:

- The first problem is finding an $[n, k, \frac{1-\varepsilon}{2}n]_2$ binary code in which the relative weight of every non-zero codeword is in the range $[\frac{1-\varepsilon}{2}, \frac{1+\varepsilon}{2}]$. Such codes are called *$\varepsilon$-balanced.*
- The second problem is finding a set $S \subseteq \{0,1\}^k$ such that for every non-empty subset $\alpha \subseteq [k]$, if we look at the random variable obtained by sampling $s = (s_1, \ldots, s_k) \in \{0,1\}^k$ uniformly at random from $S$ and outputting $\bigoplus_{i\in\alpha} s_i$, the resulting bit is almost uniform with bias at most $\varepsilon$. Such a set is called an *$\varepsilon$-biased sets.*

Finding an $\varepsilon$–balanced code is a (slightly) harder problem than finding a code with distance $\frac{1-\varepsilon}{2}$. Yet, the status of $\varepsilon$-balanced codes is similar to that of $[n, k, \frac{1-\varepsilon}{2}n]_2$ codes. For the upper bound, $n = O(\min\left\{\frac{k}{\varepsilon^2}, 2^k\right\})$ (where the probabilistic method gives non-explicit $[n,k]_2$ $\varepsilon$-balanced codes with $n = O(\frac{k}{\varepsilon^2})$, and the Hadamard code gives a perfectly balanced code with $n = 2^k$). The best lower bound is $n = \Omega(\min\left\{\frac{k}{\varepsilon^2 \log\frac{1}{\varepsilon}}, 2^k\right\})$, see [3, Section 7]. Both bounds match the corresponding bounds for $[n, k, \frac{1-\varepsilon}{2}n]_2$ codes [3]. Furthermore, the code constructions using concatenation presented above give not only codes with relative distance $\frac{1-\varepsilon}{2}$ but also $\varepsilon$–biased codes.

It is well known and easy to see that the second problem (of finding an $\varepsilon$–biased set) is *equivalent* to the first problem (of finding an $\varepsilon$–balanced code). In fact, $\varepsilon$-biased sets are just $\varepsilon$-balanced codes in a different guise: if $S$ is an $\varepsilon$–biased set over $\{0,1\}^k$ with support size $n$, then the $n \times k$ matrix $G$ that has the elements of $S$ written in its rows, is a generating matrix for an $\varepsilon$–balanced code. The opposite is also true (and easy). Thus, a construction of an $\varepsilon$–biased set over $\{0,1\}^k$ with support size $n$ is actually a construction of a generating matrix for an $[n, k, \frac{1-\varepsilon}{2}]_2$ linear code that is $\varepsilon$-balanced.

Thus, amplifying the distance of a code from relative distance $\frac{1-\varepsilon_1}{2}$ to relative distance $\frac{1-\varepsilon_2}{2}$ corresponds to amplifying the *bias* of an epsilon–biased set from $\varepsilon_1$ to $\varepsilon_2$.

There is a straightforward (and expensive) way to amplify bias. Suppose $S$ is an $\varepsilon$–biased set over $\{0,1\}^k$ with support size $n$ and let $D$ be the distribution over $\{0,1\}^k$ obtained by sampling a random element of $S$ uniformly a random. Let $D^t$ be the distribution obtained by sampling $t$ independent samples $v_1, \ldots, v_t$ from $D$ and outputting $\sum_{i=1}^{t} v_i \in \{0,1\}^k$, where the sum is in $\mathbb{F}_2^k$. It is a simple exercise to show that $D^t$ is $\varepsilon^t$-biased (see Definition 3 for the definition of bias of a distribution).

Rozenman and Wigderson suggested to replace two independent samples with pseudo-random samples [8]. Specifically, they suggested to sample an edge from an $(n, D, \lambda)$ expander, and use the two endpoints as the two samples. I.e., suppose we start with an $\varepsilon$-biased set $S$ over $\{0,1\}^k$ with support size $n$, and $G$ is an $(n, D, \lambda)$ expander with $n$ vertices. We sample an edge $(i, j) \in E$ from the expander. We then output

$z_i \oplus z_j$, where $z_\ell$ is the $\ell$'th element in the support of $S$. Rozenman and Wigderson use the expander mixing lemma to prove that the new distribution is $O(\lambda + \varepsilon^2)$ biased. Thus, if $\lambda$ is small, we have reduced the bias from $\varepsilon$ to $O(\varepsilon^2)$ while increasing the support size from $n$ to the slightly larger support size $Dn$. With this building block at hand they suggest to start with some $\varepsilon_{init}$–biased set $S_{init}$ for some constant $\varepsilon_{init}$ (e.g., Justensen code) and by repeated edge sampling, each time from a larger expander, they obtain a construction with support size about $O(\frac{k}{\varepsilon^4})$. Rozenman and Wigderson have not published their results and the results appear credited to them in Bogdanov's lecture notes of a 2012 complexity class [8].

In this paper we wish to extend this technique and bring it closer to optimal.

## 1.1 Our solution I: Long random walks over expanders are good Parity samplers

A natural suggestion extending the above results and ideas is to take a random walk of length $t$ on the expander $G$. Suppose as before that we have some $\varepsilon_{init}$–biased set $S_{init}$ over $\{0,1\}^k$ with constant bias $\varepsilon_{init}$ and linear support size $n = O(k)$. For the amplification process we use an $(n, D, \lambda)$ expander $G$ and associate the $n$ vertices of the expander with the $n$ elements in the support of $S_{init}$. The amplification itself is done by sampling a random path of length $t$ in $G$ and outputting the sum of the elements associated with vertices on the path. Specifically, if the path is $v_0, \dots, v_t \in [n]$ and $v_i$ is associated with the element $z_i \in \{0,1\}^k$ in the support of $S_{init}$, then the output is $\sum_{i=0}^t z_i \in \{0,1\}^k$ where the summation is done over $\mathbb{F}_2^k$.

But does this natural suggestion actually work?

To check this we follow the Rozenman and Wigderson framework and fix an arbitrary non-trivial test $\alpha \subseteq [k]$. The result of the test is

$$\sum_{j:j\in\alpha} (\sum_{i=0}^{t} v_i)_j \;=\; \sum_{i=0}^{t} \sum_{j:j\in\alpha} (v_i)_j,$$

where the summation is modulo 2, and $(v)_j$ is the $j$'th bit of $v \in \{0,1\}^k$. Define the set

$$B = \left\{ v \in V = [n] \mid \sum_{j:j\in\alpha} v_j = 1 \right\}.$$

We see that what we need to check is whether with probability close to half a random path of length $t$ on $G$ falls into $B$ an *odd* number of times. Indeed, does this happen?

Random walks over expanders are often used for sampling. Gilman [11] proved that as long as $\lambda$ is small enough, for every set $B \subseteq V$ the number of times we visit $B$ in a random walk of length $t$ over an expander is very close to the expectation, and the probability we deviate much from the expectation is similar to the probability $t$ completely random variables deviate from the expectation. However, in our case we are *not* interested in the number of times we fall into the set $B$ but rather in the *parity* of that number. Offhand, there is no reason to believe an expander walk is a good replacement with respect to this measure (the parity of the number of times we fall into a set).

To appreciate the above point consider what happens with pair-wise independent sampling, or even $t$–wise independent sampling. If we sample $t+1$ vertices $v_0, \dots, v_t$ $t$-wise independently from $V$, the number of times we fall into a set $B$ is indeed concentrated around the expectation. Yet, the parity may be completely biased. For example sampling $(v_0, \dots, v_t)$ such that $\sum_{i=0}^t v_i = 0$ in $\mathbb{F}_2^k$, gives a $t$-wise independent distribution, but obviously, for any test $\alpha \subseteq [k]$ we have that $\sum_{j:j\in\alpha} \sum_{i=0}^t (v_i)_j = 0$ because

$\sum_{i=0}^{t} v_i = 0$. Thus even though a random sample hits $B$ with about the right expectation, it completely fails the parity test.

Yet, doing the analysis we find out that quite surprisingly a random walk over an expander constitutes a good replacement to true randomness with regard to this parity measure, and so, informally speaking, random walks over expanders are good "parity samplers". We find this fact highly surprising.

Specifically, a random walk of length $t$ reduces the bias from $\varepsilon_{init}$ to $(\varepsilon_{init} + 2\lambda)^{t/2}$. Taking $\varepsilon_{init} \approx 2\lambda$ and $\lambda \approx \frac{2}{\sqrt{D}}$ (which is similar to the behavior of a Ramanujan graph), we see that the support size is $O(kD^t)$ while the bias is $\varepsilon = (\frac{8}{\sqrt{D}})^{t/2}$. In particular if $D$ is a large enough constant, say $D \geq 8^{1/\alpha}$, then $D^{\frac{t}{2}(\frac{1}{2}-\alpha)} \leq \frac{1}{\varepsilon}$. Therefore, the support size is of order $k \cdot D^t = \frac{k}{\varepsilon^{4+O(\alpha)}}$.

Thus, so far we have simplified the construction, and we are perhaps surprised by the proof, but nevertheless we seem to hit the same wall and we have not improved the parameters of the construction.

## 1.2 The Parity Sampling proof and why we get the $O(\frac{k}{\varepsilon^4})$ bound

We now take a closer look at the proof. As we said before, following Rozenman and Wigderson, we first fix a non-trivial test $\alpha \subseteq [k]$. The test partitions the vertices of the graph $G$ to two sets: those which index elements in the support that give 0 on the test $\alpha$, and those which give 1. Let us denote these sets by $S_0$ and $S_1$ respectively. As we explained before we need to estimate the probability a random walk on $G$ falls into $S_1$ an odd number of times.

Stated algebraically, we need to analyze the operator $(\Pi G)^t$, where $G$ is the transition matrix of the expander, and $\Pi$ is a diagonal matrix in the standard basis of the vertices, giving value 1 to vertices in $S_0$ and $-1$ to those in $S_1$. It turns out that $\|\Pi G\| = 1$, because if we let $\mathbf{1}$ be the normalized all-one vector, then $(\Pi G)\mathbf{1} = \Pi\mathbf{1}$ which is a (normalized) vector with $\pm 1$ entries, and thus has norm 1. However, $\|\Pi G \Pi G\| \leq \varepsilon_{init} + 2\lambda$ (see Theorem 15(4)) and thus one in every two steps works for us, and in $t$ steps the bias becomes $(\varepsilon_{init} + 2\lambda)^{t/2}$.

One could hope that $\left\|(\Pi G)^4\right\|$ drops down faster than $\|\Pi G \Pi G\|^2$, but it seems this might not be always the case. To see that, decompose the space to $\mathcal{V}^{\|} = \mathrm{Span}\{\mathbf{1}\}$ and its orthogonal complement $\mathcal{V}^{\perp}$, where $\mathbf{1}$ is the all-one vector (or, equivalently, the uniform distribution over $V$). Then $\mathcal{V}^{\|}$ and $\mathcal{V}^{\perp}$ are invariant under $G$ and furthermore, $G\mathbf{1} = \mathbf{1}$ while $\|Gv\| \leq \lambda\|v\|$ for every $v \in \mathcal{V}^{\perp}$. We also know how $\Pi$ acts on $\mathbf{1}$ - it moves it to a vector with $\pm 1$ entries, where the sign encodes membership in $S_1$. We do not have, however, any control on the action of $\Pi$ on $\mathcal{V}^{\perp}$.

Now consider what happens when we apply $\Pi G \Pi G$ on the vector $\mathbf{1}$. $G\mathbf{1} = \mathbf{1}$ so the first step on $G$ is wasted. Then $\Pi$ might move $\mathbf{1}$ close to a perpendicular vector (and this indeed happens because the part that is sent to $\mathcal{V}^{\|}$ is proportional to the bias and is small, see the proof of Theorem 15(4)). This is followed by the second application of $G$ which shortens the vector by $\lambda$ (because it is in the perpendicular space) and keeps it in the perpendicular space (because the perpendicular space is invariant under $G$). Finally we apply $\Pi$ again. The problem is that this second application of $\Pi$ might map the vector we currently hold close to the uniform vector $\mathbf{1}$. If this happens then, indeed, every second application of $G$ is wasted and we are doomed to get the $\frac{k}{\varepsilon^4}$ support size.

## 1.3 The replacement product

We now take a pause for the moment from our problems and take a detour to discuss the *replacement product*. Suppose we have two graphs:

- An $(n, D_1, \lambda_1)$ graph $G(V_1 = [n], E_1)$ with $n$ vertices. We think of $n$ as being large (e.g., going to infinity) while $D_1$ is relatively small, e.g., $D_1$ might be logarithmic in $n$ or a constant. We call $G$ the *outer* graph.

- A second $(D_1, D_2, \lambda_2)$ graph $H(V_2 = [D_1], E_2)$. Notice that $|V_2| = D_1$ the degree of $G$, thus $H$ is much smaller than $G$. We think of $H$ as the *inner* graph.

The replacement product $G \textcircled{r} H$ has vertex set $V_1 \times V_2$. We can visualize this as if for every vertex $v$ we replace the $D_1$ edges leaving $v$ with $D_1$ vertices of the form $(v, i)$ for $i \in V_2 = [D_1]$. We call these vertices the *cloud* of $v$. We place a copy of $H$ on each cloud. We also add an "out-going" edge from $(v, i)$ that goes where the original edge of $G$ went. To be more precise, if the $i$'th edge leaving $v$ went to $w$ and the $j$'th edge leaving $w$ is $v$ (remember that the graph is undirected) then we add an "inter-cloud" edge $((v, i), (w, j))$ to the replacement product graph.

The replacement product was analyzed in the seminal paper [19] along with the related zig-zag product. In the zig-zag product one takes the replacement product graph and replaces all edges with edges connecting the endpoints of paths of length 3 that are composed of an intra-cloud step followed by an inter-cloud step and then another intra-cloud step. In general, we will use the terminology zig-zag product when paths are shortened, and replacement product when they are not. We keep, however, the convention that a step on the replacement product is an intra-cloud step on the $H$ component followed by an inter-cloud *deterministic* step.[1]

The analysis shows that, roughly speaking, both products inherit the number of vertices from the large graph $G$, and the degree and spectral gap from the small graph $H$. It is relatively easy to construct high-degree expanders. Thus, these products reduce the task of finding an expander on many vertices to the task of finding an expander on fewer vertices, and therefore one can obtain an explicit recursive way to construct good expanders (see [19] and also [5]).

The products have added benefits beyond what one can (currently) get from other expander constructions:

- Capalbo et al. [9] used the zig-zag product to construct an explicit expander with edge expansion approaching the degree.

- Reingold [18] used the zig-zag product in the breakthrough result that undirected connectivity is in Logspace, where the zig-zag product is used to transform an arbitrary input graph to a good expander (see also [20]).

- Dinur [10] used the zig-zag product in her "combinatorial" proof of the PCP theorem.

In all these applications the zig-zag product is more than just a combinatorial replacement to an algebraic good expander, and specific properties of the product are used.

Alon et al. [4] observed a close connection between the replacement product and the semi-direct product in group theory. If $G$ and $H$ are groups and $H$ acts on $G$, one can define the semi-direct product of $G$ and

[1]In the literature it is customary to define the replacement product as the $D_1 + 1$ regular graph defined above, where each vertex has $D_1$ intra-cloud edges and one inter-cloud edge leaving (and entering) it. If we want to view this structure as a graph, forgetting about the division of the edges to two types, we need to balance the weight on inter-cloud edges to be the same as intra-cloud edges. Instead, in this paper, a random walk will remember the edge types, and walk alternately on intra-cloud and inter-cloud edges. A cleaner way to view this is that the replacement graph is a $D_1$-regular graphs, where the $i$'th neighbor of a vertex is obtained by going to the $i$'th neighbor in the cloud, and then proceeding (deterministically) over the inter-cloud edge. I.e., we "shorten" $GH$ steps.

$H$ as the group with elements $G \times H$ and product $(g_1, h_1) \cdot (g_2, h_2) = (g_1^{h_2} g_2, h_1 h_2)$. Alon et al. proved that with the appropriate choice of generators, the Cayley graph of the semi-direct product is the zig-zag product. One key feature of the semi-direct product is that the value of the $H$ coordinate is *independent* of the $G$ coordinate. This property is essential for our construction and we capture it with the condition that $G$ has a local inversion function - see Definition 9.

### 1.4 Our solution II: Protecting the de-randomized walk from the action of $\Pi$

Going back to our problem, our next attempt is to protect the instructions for the de-randomized walk from the action of $\Pi$, because we have very little control on the behavior of $\Pi$. To achieve that we use the replacement product. As before we have $G$ and $H$ and the set of vertices in the graph is $V(G) \times V(H)$. We now do the following:

- We associate the support of the $\varepsilon_{init}$–biased set $S_{init}$ with the vertices of $V(G)$. This in particular means that the linear operator $\Pi$ defined before acts on the $G$ component alone and leaves the $H$ component untouched.
- We take a $t$ long random walk on the replacement product. I.e., we start at some $v_0$. We take an intra-cloud edge, followed by the (deterministic) inter-cloud edge to get $v_1$. We repeat that $t$ times until we get the path $v_0, \ldots, v_t$. We let $z_i \in \{0,1\}^k$ be the element in the support of $S_{init}$ associated with $v_i$, and we output $\sum_{i=0}^{t} z_i$.

In the analysis we now have *three* operators: $H$ that acts on the $H$ component and describes the intra-cloud dynamics, $G$ that is a deterministic step (in fact a permutation on the vertices) and describes the inter-cloud step, and $\Pi$ that acts on the $G$ component alone and changes the sign according to membership in $B$. A step in the $t$ long random walk corresponds to $\Pi G H$, i.e., first an intra-cloud step, then the inter-cloud deterministic step, and then $\Pi$ gives a $\pm 1$ value based on the result of the test $\alpha$.

Doing the analysis we get the same parameters as before (in fact, slightly worse parameters, as we cannot associate the vertices of $H$ with the $\varepsilon_{init}$ biased-set $S_{init}$). The analysis, again, shows one in every two steps can be wasted. To see that we let $\mathcal{V}^{\|}$ be the vector space containing all vectors that are uniform inside each cloud (algebraically this means the vector is a product vector $v_1 \otimes v_2$ with $v_2$ being uniform over $V_2 = V(H)$) and $\mathcal{V}^{\perp}$ be its orthogonal complement. A bad example is, e.g., a parallel vector in $\mathcal{V}^{\|}$. On a parallel vector the first $H$ step is wasted and $\Pi G$ moves the vector close to the perpendicular space (intuitively this happens because the part that is sent to $\mathcal{V}^{\|}$ is proportional to the bias and is small. This, of course, requires a proof and is a special case of Theorem 27). The second $H$ application shortens the vector by $\lambda_2$ and keeps it in the perpendicular space, then the second $\Pi G$ step might move the vector close to a parallel vector in $\mathcal{V}^{\|}$.

At first, this looks very similar to the situation we had before and it seems we have gained nothing. However, this is not necessarily true. This is because $\Pi$ is almost entirely out of our control (it is determined by the test $\alpha$ and there are about $2^k$ possible tests). In contrast, $G$ and $H$ are chosen by the algorithm. Thus, the fact that $\Pi$ does not meddle with the $H$ component (which is what determines the next step on the path) gives us, at least potentially, the hope that our situation would be better if we choose $G$ and $H$ appropriately.

In fact Ben-Aroya et al. [6] had to solve exactly the same problem, but in a situation without the $\Pi$ operator. For that [6] defined the $s$-wide walk. Applying the same technique in our more complicated scenario (with the $\Pi$ operator) indeed works *as long as* $\Pi$ *does not act on the* $H$ *vertex.* Thus, the main purpose of using the replacement product here is to separate between the action of $\Pi$ and the pseudo-random machinery that determines the path.

We now explain the technique used by [6] and how it is applied in our case to give the improved result we seek.

## 1.5 Our solution III: Using an $s$-wide replacement product walk

We now use a technique developed in Ben-Aroya et al. [6]. The idea, informally speaking, is to make sure that if a certain step in the random walk does not work for us, then many following steps work well. If we succeed in that and can arrange, say, that in every $s$ steps $s - O(1)$ steps work for us, then we should get an $\varepsilon$–biased set with support size $\frac{k}{\varepsilon^{2+O(\frac{1}{s})}}$.

To make the idea work we first notice that each step costs $D_2$ (because the inter-cloud steps are deterministic) and may reduce the norm by $\lambda_2$, and both $D_2$ and $\lambda_2$ are independent of $|V_2|$. Thus, we could have made $V_2$ larger, seemingly without changing the cost.[2] Thus, we choose to make $|V_2| = D_1^s$ for some parameter $s$ (and for a first reading we recommend the reader to think of $s$ as a large constant) and we think of an element $v \in V_2 = [D_1]^s$ as having $s$ blocks $v_1, \ldots, v_s$ each being a $[D_1]$ instruction. We need to associate elements $v = (v_1, \ldots, v_s) \in V_2 = [D_1]^s$ with labels in $[D_1]$ and for concreteness let us say that at step $i$ the label $\pi_i(v)$ associated with $v$ is determined by the element $v_{i \bmod s} \in [D_1]$. We call this kind of random walk the *s–wide random walk on the replacement product graph.*

We can carry on the same analysis as before. Again, $\|\Pi G\| = 1$ and the parallel vectors $v \otimes \mathbf{1}$ are problematic elements. The crucial point is that if we are in a parallel vector, then we are uniform on the cloud and therefore have $\log(|V_2|) = s \log(D_1)$ uniform bits. In theory, we could have hoped that $H$ mixes the bits so well that the labels we get in the next $s$ steps are completely *uniform and independent*. Ben-Aroya et al. prove that this is indeed the case, w.h.p., with a random degree $D_2$ graph $H$. Thus, if we fail once, we can expect the next $s' = s - O(1)$ steps form a perfect length $s'$ random walk on $G$! Notice that each step only costs $\log D_2$ bits, but buys us a perfect random walk on $G$ that worths $\log(D_1) \gg \log(D_2)$ bits. Hence when we get to be in a parallel vector we indeed lose an $H$ step, but we gain much more from the fact that we invest $\log D_2$ bits and they worth us $\log D_1$ bits. The technical instantiation of this is Theorem 27 about the action of the operator on parallel vectors.

If we resort to finding $H$ by a brute force search we have to take $H$ to be constant sized (or slightly more) and we get support size $\frac{k}{\varepsilon^{2+\alpha}}$ for arbitrarily small constant $\alpha$. An anonymous referee to the STOC 2008 submission of [6] told us how to replace the brute-force search for a good $H$ with an explicit construction. The idea is to take $H$ to be a Cayley graph over an Abeliean group $\mathbb{Z}_{D_1}^s$, and do the walk on the $s$-wide product $G \textcircled{r} H$ such that in the $i$'th step we walk according to the $(i \mod s)$ coordinate of the $V_2 = \mathbb{Z}_{D_1}^s$ element (see Section 2.4). The fact that the Cayley graph respects the vector space structure of $\mathbb{Z}_{D_1}^s$ translates to a simple proof that $H$ is good for us (see Section 7). The only potential drawback is that Abelian Cayley graphs require poly-logarithmic degree, but this is insignifcant in our setting. Working with a nicely structured $H$ simplifies the construction, makes it more concrete and also enables us to use larger graphs $H$ thereby optimizing the parameters better. We are grateful for the anonymous referee for his/her suggestions.

## 1.6 The parameters we get

Taking the approach outlined above we prove Theorem 2. To get some intuition about the parameters notice that the support size is $|V_1| \cdot |V_2| \cdot D_2^t = n \cdot D_1^s \cdot D_2^t$ which is dominated by $n \cdot D_2^t = O(kD_2^t)$. We want to make this quantity close to $\frac{k}{\varepsilon^2}$ and so we want $D_2^t$ to be as close as possible to $\frac{1}{\varepsilon^2}$. As we explained before,

[2]In our application we associate the $n$ elements with $V_1$ alone, so we "lose" $V_2$, which means that we cannot make $V_2$ as large as we wish. Still, we do not pay much for making $V_2$ larger.

in an ideal world (that does not exist) we would lower the norm by $\lambda_2$ at each of the $t$ steps. If, furthermore, $\lambda_2$ would have been $\frac{1}{\sqrt{D_2}}$ (which is again impossible) we would get a perfect result since $\varepsilon$ would be $(\frac{1}{\lambda_2})^t$ which would be $D_2^{-t/2}$, and then $D_2^t$ would be $\frac{1}{\varepsilon^2}$.

Although the perfect world does not exist we get close to it with some imperfections:

1. $\lambda_2$ is not $\frac{1}{\sqrt{D_2}}$ but rather $\frac{2}{\sqrt{D_2}}$ or larger. This is unavoidable since the number of vertices in $H$ is $D_1^s \geq D_2^s$ which is much larger than the degree $D_2$ [16]. This forces $D_2$ to be large enough so that the factor of 2 is tiny, and this is the reason why we need to enforce $D_2 \geq 2^s$ (and, in fact, we need $D_2$ to be slightly larger).

2. Another imperfection is that we put the elements in the support of $S_{init}$ over the vertices $V_1$ of $G$ alone, while the graph has vertex set $V_1 \times V_2$, and the support size of $\Upsilon$ is $|V_1 \times V_2| \times D_2^t$. Thus we lose a factor of $|V_2| = D_1^s \geq D_2^s$ in the support size. This forces us to take s small compared to $t$, and is the reason why we enforce $t \geq \frac{s}{\alpha}$.

3. Finally, when we take a walk of length $s$ not all $s$ steps "work" for us, and we can prove only that $s-4$ work for us. In fact, if we look at $s$ steps alone, there are always scenarios where we lose at least one step, so it seems we cannot hope to improve the $s-4$ to anything better than $s-1$. This forces 1 (or 4) to be tiny compared to $s$, and is the reason why we enforce $s \geq \frac{1}{\alpha}$.

Putting this together, and roughly speaking, $D_2 \geq 2^s$ and $D_2^t \geq 2^{st} \geq 2^{s^2/\alpha} \geq 2^{1/\alpha^3}$. $D_2^t$ should also be poly($\frac{1}{\varepsilon}$). Thus we see that we should expect something like $\log\frac{1}{\varepsilon} = \frac{1}{\alpha^3}$, i.e., $\alpha = \frac{1}{(\log\frac{1}{\varepsilon})^{1/3}}$. Indeed, the actual bound that we prove is not far from that and gives $\alpha = \Theta((\frac{\log\log\frac{1}{\varepsilon}}{\log\frac{1}{\varepsilon}})^{1/3})$.

Specifically, We prove:

**Theorem 1.** *For every constant $\alpha > 0$, there exists an explicit construction that given $k$ and $\varepsilon > 0$ constructs an $\varepsilon$–biased set over $\{0,1\}^k$ with support size $O(\frac{k}{\varepsilon^{2+\alpha}})$.*

If we want to be precise about the parameters we get, then:

**Theorem 2.** *There exists an explicit construction that given $k$ and $\varepsilon > 0$ constructs an $\varepsilon$–biased set over $\{0,1\}^k$ with support size*

$$k \cdot 2^{2\log\frac{1}{\varepsilon} + O((\log\frac{1}{\varepsilon})^{2/3}(\log\log\frac{1}{\varepsilon})^{1/3})} \tag{1}$$

*Said differently, the support size is $n = O(\frac{k}{\varepsilon^{2+\alpha}})$ for $\alpha = O(\frac{(\log\frac{1}{\varepsilon})^{2/3}(\log\log\frac{1}{\varepsilon})^{1/3}}{\log\frac{1}{\varepsilon}}) = O((\frac{\log\log\frac{1}{\varepsilon}}{\log\frac{1}{\varepsilon}})^{1/3})$.*

We remark that when $\varepsilon \leq \frac{1}{k\sqrt{\log k}}$, the construction of [3] that has support size $O(\frac{k^2}{\varepsilon^2})$ already achieves the support size stated in Eq (1). Hence we can assume, w.l.og., that $\varepsilon \geq \frac{1}{k\sqrt{\log k}}$. For a first reading we recommend the reader to think of $\varepsilon$ as being some polynomial in $\frac{1}{k}$.

## 1.7 Some concluding remarks and thoughts

$s$-wide random walks were introduced in Ben-Aroya et al. [6]. The walk in [6] is of length exactly $s$ and the walk is shortened to the two endpoints of the path as is done in the zig-zag product. [6] show the resulting graph is almost Ramanujan. In this work we do not shorten the path and instead look at the coherent walk on the replacement product graph (see also footnote 1).[3] We also consider longer walks, and allow intermediate operations on a vertex (like the operator $\Pi$ in our case) as long as they do not break the independence of the $H$ coordinate. We believe this generalized setting is quite natural, and we expect to see more applications where such random walks are useful.

Looking back at the construction outlined above, it seems the key property allowing the existence of a good $H$ (in both the existential proof and the explicit construction) is the fact that in our construction the $H$ coordinate is "protected" from the action of both $G$ and $\Pi$. This is manifested in the requirement that $G$ has a local inversion function (to protect $H$ from the action of $G$) and the choice to associate the support of the $\varepsilon_{init}$–biased set $S_{init}$ with $V_1$ alone (and thus $\Pi$ ignores the $H$ coordinate). This key property is also featured in the semi-direct product where $H$ evolves independently of the $G$ coordinate. We are not aware of any previous use of such "semi-direct" random walks.

We conclude with an intriguing open problem. As we explained before an explicit $\varepsilon$-biased construction over $\{0,1\}^k$ with support size $n$ is equivalent to the explicit construction of the *generating matrix* of a $[n, k, \frac{1-\varepsilon}{2}]_2$ balanced code. Thus, we have explicitly constructed the generating matrix of an error-correcting code that has distance close to half and almost optimal rate. We stress, however, that our construction does not give an explicit *decoding algorithm*. Finding such a decoding algorithm is a natural important open problem.

## 2 Preliminaries

### 2.1 Epsilon-biased sets

**Definition 3.** *($\varepsilon$-bias) Let $\Upsilon$ be a distribution over $\{0,1\}^k$. For $\alpha \in \{0,1\}^k$ define:*

$$Bias_\alpha(\Upsilon) \quad = \quad \left| \Pr_{s \in \Upsilon}(\langle \alpha, s\rangle = 0) - \Pr_{s \in \Upsilon}(\langle \alpha, s\rangle = 1) \right|,$$

*where the inner product is taken over $\mathbb{F}_2$. We call $\alpha$ a* linear test. *Denote $Bias(\Upsilon) = \max_{\alpha \neq \emptyset} Bias_\alpha(\Upsilon)$. We say $\Upsilon$ is $\varepsilon$—biased if $Bias(\Upsilon) \le \varepsilon$.*

We can identify an element $\alpha \in \{0,1\}^k$ with the set $\{i \in [k] \mid \alpha_i = 1\}$. In this terminology,

$$Bias_\alpha(\Upsilon) \quad = \quad \left| \mathbb{E}_{s \in \Upsilon} (-1)^{\sum_{i \in \alpha} s_i} \right|.$$

**Definition 4.** *($\varepsilon$–biased set) A multi-set $S$ over $\{0,1\}^k$ with support size $n$ is a function $S : [n] \to \{0,1\}^k$. $S$ is $\varepsilon$-biased if the distribution $\Upsilon = S(U_n)$, obtained by picking $i \in [n]$ uniformly at random and outputting $S(i)$, is $\varepsilon$–biased. A family $\{S_k\}_{k \in \mathbb{N}}$ is $(n, k, \varepsilon)$–biased if for every $k$, $S_k : [n = n(k)] \to \{0,1\}^k$ is $\varepsilon = \varepsilon(k)$ biased. We call $S : [n] \to \{0,1\}^k$ the* index *function of $\Upsilon$.*

**Definition 5.** *(explicit and fully explicit sets) A family $\{S_k\}_{k \in \mathbb{N}}$ of $S_k : [n] \to \{0,1\}^k$ is* explicit*, if there exists an algorithm that outputs all elements in the support of $S_k$ in time poly$(n, k)$. The family is* fully explicit *if there exists a polynomial time algorithm that on input $k$, $i \in [n]$ and $j \in [k]$, outputs the $j$'th bit of $S_k(i)$ in polynomial time in the input length, i.e., in time poly$(\log(n + k))$.*

[3]To be precise, this is true except that the walk also needs to remember the index mod $s$ of the step.

As we said before, $\varepsilon$-biased sets are just $\varepsilon$-balanced codes in a different guise: the rows of a matrix whose columns generate an $\varepsilon$-balanced code form an $\varepsilon$-biased set, and vice versa. In terms of parameters, an $[n,k]_2$ $\varepsilon$-balanced code is equivalent to an $\varepsilon$-biased set $S \subseteq \{0,1\}^k$ of size $n$. Thus, in error correcting codes terminology, *explicit* means one can output the generating matrix of the code in time $poly(n)$, while *fully explicit* means one can output each entry of the generating matrix in time poly-logarithmic in the dimension.[4]

We use several basic $\varepsilon$–biased constructions. One construction is AGHP with slightly simplified bounds, mainly because we do not care about logarithmic factors:

**Lemma 6.** *(AGHP) [3] For every $\varepsilon = \varepsilon(k)$ there exists a fully explicit family $\{S_k\}$ that is $(n,k,\varepsilon)$–biased with support size $n \leq \frac{2k^2}{\varepsilon^2}$.*

Another construction we need is of a linear size, constant bias set:

**Lemma 7.** *(Justensen) [13] For every constant $\alpha < H^{-1}(\frac{1}{2}) \approx 0.11$ and every constant $c$ large enough, there exists a fully explicit family $\{S_k\}$ that is $(n = ck, k, 1-2\alpha)$–biased. Notice the linear support size $n = ck$.*

We prove both lemmas in Section 8, mainly because of the explicitness issue.

We also need an explicit $\varepsilon$–biased set with support size $O(\frac{k}{\text{poly}(\varepsilon)})$. We could have used [15] with support size about $\frac{k}{\varepsilon^3}$, but instead we use the construction with a random walk over expanders. There are three reasons for that: first, the construction is simpler, second, it is a good preparation for the construction we present, and third, we need the random-walk case for analyzing the s-wide walk.

**Theorem 8.** *There exists a constant $c$ such that for every $\varepsilon = \varepsilon(k)$ there exists an explicit family $\{S_k\}$ that is $(n,k,\varepsilon)$–biased with support size $n \leq O(\frac{k}{\varepsilon^c})$.*

We remark that the constant $c$ can be made as close as we wish to 4, but this is not essential to us. We present the proof in Section 3. We also remark we only claim explicitness rather than full explicitness. We discuss this issue in Sec 8.3.1.

## 2.2 Rotation maps.

Following [19] we represent undirected regular graphs using *rotation maps* as we explain now. Let $G$ be an undirected $D$–regular graph $G = (V,E)$. We assume every vertex $v$ has a labeling of the $D$ edges adjacent to it with the labels $\{1,\ldots,D\}$ such that every label appears once. Notice that each edge $(a,b)$ is labeled twice, once by $a$ and once by $b$, and these labels might be different. Let $v_G[i]$ denote the $i$'th neighbor of $v$ according to the labeling of $G$. The labeling induces what is a called *the rotation map* of the graph $G$, $\mathrm{Rot}_G : V \times [D] \to V \times [D]$, defined by

$$\mathrm{Rot}_G(v,i) = (w,j) \quad \Longleftrightarrow \quad v_G[i] = w \text{ and } w_G[j] = v.$$

In words, the $i$'th neighbor of $v$ in $G$ is $w$, and the $j$'th neighbor of $w$ in is $v$. Notice that if $\mathrm{Rot}_G(v,i) = (w,j)$ then $\mathrm{Rot}_G(w,j) = (v,i)$, so Rot is a permutation on $V \times [D]$ (in fact, an involution).

We single out a special family of rotation maps:

**Definition 9.** *A graph $G$ is* locally invertible *if there exists a permutation $\phi : [D] \to [D]$ such that $\mathrm{Rot}_G(v,i) = (v[i], \phi(i))$. We say that $\phi$ is the local inversion function.*

[4]Sudan, in his lecture notes [22], calls this property "locally poly-time constructible".

For example, suppose $G$ is the 3-cycle with $V = \{v_1, v_2, v_3\}$ and $E = \{(v_1, v_2), (v_1, v_3), (v_2, v_3)\}$. It is easy to see that there is no labeling of the edges such that for each vertex the two edges adjacent to it are labeled differently. The labeling $v_1[1] = v_2, v_1[2] = v_3, v_2[1] = v_1, v_2[2] = v_3, v_3[1] = v_2$ and $v_3[2] = v_1$ is a valid labeling, but it is not locally invertible. The labeling $v_1[1] = v_2, v_2[1] = v_3, v_3[1] = v_1, v_1[2] = v_3, v_2[2] = v_1$ and $v_3[2] = v_2$ is valid and locally labeled with the local inversion function $\phi(1) = 2$ and $\phi(2) = 1$.

Let $G$ be a group and $S \subseteq G$ a set closed under inverse. The *Cayley* graph $Cay(G, S)$ is the undirected graph where the vertices of the graph are the elements of the group $G$ and $(a, b) \in E$ iff $ab^{-1} \in S$. For example, if the group is $\mathbb{Z}_2^n$ and $S = \{e_1, \ldots, e_n\}$ where $e_i$ has one in the $i$'th coordinate and zero otherwise, then $Cay(\mathbb{Z}_2^n, S)$ is the graph of the boolean cube. If $G = (\mathbb{Z}_3, +)$ and $S = \{\pm 1\}$ we get the example above of the graph with 3 vertices.

The Cayley graph $Cay(G, S)$ is regular with $|G|$ vertices and degree $D = |S|$. Every such a Cayley graph has a natural locally invertible mapping where vertex $a$ labels the edge $(a, b)$ going into $b$ with the label $a^{-1}b \in S$. It is a simple check that with this labeling the local inversion function $\phi : S \to S$ is $\phi(s) = s^{-1}$.

## 2.3 Expanders and spectral gap.

We associate a $D$-regular graph $G = (V, E)$ with its transition matrix which we also denote by $G$, i.e., $G_{v,u} = \frac{1}{D}$ if $(v, u) \in E$ and 0 otherwise. Since $G$ is regular and undirected its transition matrix $G$ is Hermitian and $G$ has an orthonormal eigenvector basis with real eigenvalues $\lambda_1 \geq \ldots \geq \lambda_N$. Also, since $G$ is regular $\lambda_1 = 1$. We denote,

$$\bar{\lambda}(G) = \max\{\lambda_2, -\lambda_N\}.$$

We say an undirected, $D$–regular graph $G$ is *Ramanujan* if

$$\bar{\lambda}(G) \leq \lambda_{\mathrm{Ram}}(D) \stackrel{\mathrm{def}}{=} \frac{2\sqrt{D-1}}{D}.$$

Ramanujan graphs are essentially optimal algebraic expanders [16].

We will need *explicit* graphs:

**Definition 10.** *(explicit graph) A family of graphs $\{G_n\}$ is a $(n, D(n), \lambda(n))$ family, if $G_n$ has $n$ vertices, degree $D(n)$ and $\bar{\lambda}(G_n) \leq \lambda(n)$. We say $\{G_n\}$ is* explicit*, if there exists an algorithm that on input $1^n$ outputs $G_n$ in time poly$(n)$. $\{G_n\}$ is* fully explicit *if there exists an algorithm that given $n, v \in [n], i \in [D(n)]$ outputs $v[i]$ in time poly$\log n$.*

Cayley graphs are often used for (fully) explicit constructions of expanders. If the group $G$ is Abelian and the set of generators $S$ is small, then the graph $Cay(G, S)$ cannot be a good expander. However, when high degree is allowed commutativity is not a big problem. It is easy to analyze the spectrum of a Cayley graph over an Abelian group: it is a simple and well known fact that the eigenvectors of the graph are the characters of the group $G$, and, in particular, the eigenvectors do not depend on $S$. From that it is easy to see that:

**Lemma 11.** *For every $S \subseteq \mathbb{Z}_2^n$, $\bar{\lambda}(Cay(\mathbb{Z}_2^n, S)) = Bias(S)$. In particular, $Cay(\mathbb{Z}_2^n, S)$ is a $(N = 2^n, D = |S|, \lambda = Bias(S))$ graph.*

There are fully explicit constructions of Ramanujan Cayley graphs over non-Abelian groups. In particular, for every $D = p^k + 1$, where $p$ is prime and $k$ is integer, there exists a fully explicit family of degree

*D Ramanujan* graphs (see, e.g., [14, 12]). However, we will need a family with varying degree and spectral gap. This, in a sense, makes the construction easier because we allow a larger degree, but also puts more burden on the explicitness of the construction because, e.g., in a fixed degree construction with a Cayley graph one can fix the generators, whereas, when $D$ varies with the input length the generators also need to be explicit. Yet, the constructions in [14] can essentially handle this situation too, giving explicit (but not necessarily fully explicit) constructions. Formally,

**Lemma 12.** *[14] For every $\beta > 0$, there exists an algorithm that given $n$ and $0 < \bar{\lambda} < 1$ runs in time $poly(n)$ and outputs a Ramanujan graph $G$ with $D \leq \frac{8}{\bar{\lambda}^2}$ and $\bar{\lambda}(G) \leq \bar{\lambda}$. The number of vertices of $G$ is either in the range $[(1-\beta)n, n]$ or in the range $[(1-\beta)2n, 2n]$. The algorithm also outputs a local inversion function $\phi : [D] \to [D]$ computable in polynomial time in its input length.*

We explain Lemma 12 in Section 8.

## 2.4 The $s$-wide replacement product

The input to the product is:

- An undirected graph $G = (V_1 = [N_1], E_1)$ that is a $(N_1, D_1, \lambda_1)$ graph. We assume $G$ has a local inversion function $\phi = \phi_G : [D_1] \to [D_1]$. That is, $\mathrm{Rot}_G(v^{(1)}, d_1) = (v^{(1)}[d_1], \phi_G(d_1))$.
- An undirected graph $H$ that is a $(N_2 = D_1^s, D_2, \lambda_2)$ graph over the vertex set $V_2 = [N_2]$.

In the replacement product the parameters are set such that the cardinality of $V_2$ equals the degree $D_1$ of $G$. An element $v_2 \in V_2$ is then interpreted as a label $d_1 \in [D_1]$. However, as explained in the introduction, we take a larger graph $H$ with $V_2 = [D_1]^s$. That is, we have $D_1^s$ vertices in $V_2$ rather than $D_1$ in the replacement product. Therefore, we need to explain how to map a vertex $v^{(2)} \in V_2 = [D_1]^s$ to a label $d_1 \in [D_1]$ of $G$. For that we let $\pi_i : V_2 = [D_1]^s \to [D_1]$, for $i = 0, \ldots, s-1$, be the map

$$\pi_i(v_0^{(2)}, \ldots, v_{s-1}^{(2)}) \quad = \quad v_i^{(2)}$$

I.e., $\pi_i$ projects $v^{(2)} \in [D_1]^s$ to the $i$'th coordinate.

**Definition 13.** *Fix $t$. Let $v \in [N_1] \times [N_2]$ and $\bar{i} = (i_0, \ldots, i_{t-1}) \in [D_2]^t$. The path indexed by $\bar{i}$ starting at $v$ is $(v_0, \ldots, v_t) \in ([N_1] \times [N_2])^{t+1}$ where:*

- $v_0 = v = (v^{(1)}, v^{(2)})$.
- *For $\ell = 0, \ldots, t-1$,*
  - *Take one step on $H$, i.e.,*

$$\begin{aligned} temp_\ell^{(1)} &= v_\ell^{(1)}, \textit{ and,} \\ temp_\ell^{(2)} &= v_\ell^{(2)}[i_\ell]. \end{aligned}$$

*Notice that we leave the first component untouched.*

– *Take one step on G with $\pi_{\ell \bmod s}(temp_\ell^{(2)})$ as the $[D_1]$ label to be used on G. I.e.,*

$$\begin{aligned} v_{\ell+1}^{(1)} &= v_\ell^{(1)}[\pi_{\ell \bmod s}(temp_\ell^{(2)})], \text{ and,} \\ v_{\ell+1}^{(2)} &= \psi_{\ell \bmod s}(temp_\ell^{(2)}), \end{aligned}$$

*where for $0 \le j \le s-1$,*

$$\psi_j(v^{(2)}) = (\pi_0(v^{(2)}), \ldots, \pi_{j-1}(v^{(2)}), \phi_G(\pi_j(v^{(2)})), \pi_{j+1}(v^{(2)}), \ldots, \pi_{s-1}(v^{(2)})). \quad (2)$$

*Namely, $\psi_j$ keeps all coordinates unchanged, except for the $j$'th coordinate where it applies the local inversion function of G.*

*We write $PATH(v, \bar{i}) = (v_0, \ldots, v_t)$.*

To summarize, we start with a locally invertible, $D_1$–regular graph over $N_1$ vertices. We replace each degree $D_1$ vertex with a "cloud" of $D_1^s$ vertices, and map a cloud vertex to a $D_1$ instruction at step $\ell$ using $\pi_{\ell \bmod s}$. We take a $2t$-step walk, with alternating $H$ and $G$ steps.

## 2.5 Miscellaneous notation.

For an $n$-dimensional vector $x$ we let $|x|_1 = \sum_{i=1}^n |x_i|$ and $\|x\| = \sqrt{\langle x, x\rangle}$. If $V$ is a set and $\mathcal{V} = \mathrm{Span}(V)$ with $\overrightarrow{v}$ the basis vector corresponding to $v$, we let $\mathbf{1}_\mathcal{V}$ be the all-ones vector over $V$ normalized to unit length, namely $\mathbf{1}_\mathcal{V} = \frac{1}{\sqrt{|V|}} \sum_{v \in V} \overrightarrow{v}$. When the vector space is clear from the context we simply denote this vector by $\mathbf{1}$.

We often use vectors coming from a tensor vector space $\mathcal{V} = \mathcal{V}_1 \otimes \mathcal{V}_2$, as well as vertices coming from a product vertex set $V = V_1 \times V_2$. In such cases we use superscripts to indicate the universe a certain object resides in. For example, we denote vectors from $\mathcal{V}_1$ by $x^{(1)}, y^{(1)}$ etc. In particular, when $x \in \mathcal{V} = \mathcal{V}_1 \otimes \mathcal{V}_2$ is a product vector then $x^{(1)}$ denotes the $\mathcal{V}_1$ component, $x^{(2)}$ denotes the $\mathcal{V}_2$ component and $x = x^{(1)} \otimes x^{(2)}$.

As in the analysis of the zig-zag product, we decompose $\mathcal{V} = \mathcal{V}_1 \otimes \mathcal{V}_2$ to its parallel and perpendicular parts. The subspace $\mathcal{V}^{\|}$ is defined by

$$\mathcal{V}^{\|} = \mathrm{Span}\left\{ \overrightarrow{v^{(1)}} \otimes \mathbf{1} : v^{(1)} \in V_1 \right\}$$

and $\mathcal{V}^{\perp}$ is its orthogonal complement. For any vector $\tau \in \mathcal{V}$ we denote by $\tau^{\|}$ and $\tau^{\perp}$ the projections of $\tau$ on $\mathcal{V}^{\|}$ and $\mathcal{V}^{\perp}$ respectively. Notice that $\mathcal{V}^{\|}$ is exactly the set of parallel vectors defined in the introduction, and $\mathcal{V}^{\perp}$ is the set of perpendicular vectors. Also notice that $v \in \mathcal{V}^{\|}$ iff $v = v_1 \otimes \mathbf{1}$ for some $v_1 \in \mathcal{V}_1$.

$\mathbb{S}_\Lambda$ denotes the symmetric group over $\Lambda$.

We measure the distance between two distributions $P, Q$ by $|P - Q|_1$. The operator norm of a linear operator $L$ is $\|L\|_\infty = \max_{x:\|x\|=1} \|Lx\|$. We will need the following claim which states that if we average linear operators according to two statistically-close distributions, we get essentially the same linear operator.

**Claim 14.** *Let $P, Q$ be two distributions over $\Omega$ and let $\{\mathcal{L}_i\}_{i \in \Omega}$ be an arbitrary set of linear operators over $\Lambda$ each with operator norm bounded by 1. Define $\mathcal{P} = \mathbb{E}_{x \sim P}[\mathcal{L}_x]$ and $\mathcal{Q} = \mathbb{E}_{x \sim Q}[\mathcal{L}_x]$. Then, for any $\tau \in \Lambda$,*

$$\|\mathcal{P}\tau - \mathcal{Q}\tau\| \le |P - Q|_1 \cdot \|\tau\|.$$

**Proof:** The claim follows from

$$\|\mathcal{P}-\mathcal{Q}\|_\infty \;=\; \left\|\sum_{x\in\Omega}(P(x)-Q(x))\mathcal{L}_x\right\|_\infty \;\le\; \sum_x |P(x)-Q(x)|\cdot\|\mathcal{L}_x\|_\infty \;\le\; |P-Q|_1.$$

■

## 3 Amplifying bias with a random walk over an expander

### 3.1 The construction

The input to the construction is $k$ and $\varepsilon$. We fix $\varepsilon_0 = 0.8$, $\beta = 0.01$ and $\lambda$ such that $\zeta = \varepsilon_0 + 2\beta + 2\lambda < 0.9$. Then choose:

**A distribution $\Upsilon_0$ with constant bias** : We use Lemma 7 to get an $\varepsilon_0 = 0.8$ biased distribution $\Upsilon_0$ over $\{0,1\}^k$ and support size $n$. We calculate this $n$. Let $Z : [n] \to \{0,1\}^k$ be the index function of $\Upsilon_0$.

**An expander graph** : We use Lemma 12 with $n$ and $\lambda$ to find an explicit undirected graph $G$ that is a $(n', D, \lambda)$ Ramanujan expander, with $\bar{\lambda}(G) \le \lambda$ and $D \le \frac{8}{\lambda^2}$. The number of vertices $n'$ is either in the interval $[(1-\beta)n, n]$ or in the interval $[(1-\beta)2n, 2n]$. W.l.o.g., it's the first case and we let $\Upsilon_0$ be the $\varepsilon_0$–biased distribution obtained by Lemma 7 with support size $n$ (otherwise we duplicate every element twice to get support size $2n$). Notice that $n'$ which is the number of vertices in the graph $G$ almost matches $n$ which is the support size of the $\varepsilon_0$–biased distribution. Specifically, $\delta = \frac{n-n'}{n'} \le 2\beta n'$.

**Walk length** : We fix $t = 2\lceil \log_\zeta \varepsilon \rceil$.

The distribution $\Upsilon$ is obtained as follows:

1. Sample $v \in [n']$ and $\bar{i} = (i_0, \ldots, i_{t-1}) \in [D_2]^t$. Let $PATH(v, \bar{i}) = (v_0, \ldots, v_t)$ be the path obtained by starting at $v_0$ and walking over $G$ according to the instructions $\bar{i}$.
2. Output $Z(v_0^{(1)}) \oplus \ldots \oplus Z(v_t^{(1)}) \in \{0,1\}^k$.

Obviously $\Upsilon$ is distributed over $\{0,1\}^k$ and has support size $n'D_2^t \le nD_2^t$. Next, we will see that $\Upsilon$ is $\varepsilon$–biased and prove Theorem 8.

### 3.2 Expressing the bias algebraically

We want to express the walk as a composition of linear operators. We define a vector space $\mathcal{V}$ with $\dim(\mathcal{V}) = |V| = n'$ and identify an element $v \in V$ with a basis vector $\overrightarrow{v} \in \mathcal{V}$. On this basis we define the linear operator $G$ which is the operator corresponding to a random walk on $G$.

Next, let $\alpha = (\alpha_1, \ldots, \alpha_k) \in \{0,1\}^k$ be a non-trivial linear test maximizing the bias of $\Upsilon$. Let $S_0$ and $S_1$ be the partition of $[n']$ associated with it,

$$S_b = \big\{ v \in [n'] \mid \langle Z(v) | \alpha \rangle = b \big\}.$$

Define $\Pi_0$ and $\Pi_1$ where $\Pi_b$ is the projection on the vector space $Span(\{\overrightarrow{v} \mid v \in S_b\})$. We let

$$\Pi = \Pi_0 - \Pi_1.$$

Finally, the construction uses a random walk where the path $(v_0, \dots, v_t)$ is sampled according to the distribution $PATH(U_{[n']}, U_{[D_2^t]})$. Let $p_{even}(S_1)$ (respectively $p_{odd}(S_1)$) be the probability a sampled path $(v_0, \dots, v_t)$ visits $S_1$ an even number (respectively odd number) of times. With this notation:

**Theorem 15.** *We have:*

1. $Bias(\Upsilon) = Bias_\alpha(\Upsilon) = |p_{even}(S_1) - p_{odd}(S_1)|$,
2. $p_{even}(S_1) - p_{odd}(S_1) = \sum_{b_0,\dots,b_t \in \{0,1\}} (-1)^{b_0+\dots+b_t} \mathbf{1}^\dagger \Pi_{b_t} G \Pi_{b_{t-1}} \dots \Pi_{b_2} G \Pi_{b_1} G \Pi_{b_0} \mathbf{1}$,
3. $p_{even}(S_1) - p_{odd}(S_1) = \mathbf{1}^\dagger (\Pi G)^t \Pi \mathbf{1}$,
4. $\left\|(\Pi G)^2\right\| \le \varepsilon_0 + 2\beta + 2\lambda$, *and,*
5. $Bias(\Upsilon) \le (\varepsilon_0 + 2\beta + 2\lambda)^{\lfloor t/2 \rfloor}$.

**Proof:**

**Item 1.** By definition of $\alpha$, $Bias(\Upsilon) = Bias_\alpha(\Upsilon)$. Also,

$$\begin{aligned} Bias_\alpha(\Upsilon) &= |\mathbb{E}_{z \in \Upsilon} (-1)^{\langle z, \alpha \rangle}| = |\mathbb{E}_{(v_0,\dots,v_t) \in PATH(U_{[n']}, U_{[D_2^t]})} (-1)^{\left\langle \oplus_{j=0}^t Z(v_j), \alpha \right\rangle}| \\ &= |\mathbb{E}_{v_0,\dots,v_t} (-1)^{\oplus_{j=0}^t \langle Z(v_j), \alpha \rangle}| \\ &= |p_{even}(S_1) - p_{odd}(S_1)|. \end{aligned}$$

**Item 2.** First, $\frac{1}{|V|} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ corresponds the uniform distribution over $V = [n']$. Therefore,

$$\begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}^\dagger \Pi_{b_t} G \Pi_{b_{t-1}} \dots \Pi_{b_2} G \Pi_{b_1} G \Pi_{b_0} \frac{1}{|V|} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$$

is the probability that $v_j \in S_{b_j}$ for $j = 0, \dots t$. The whole sum is $p_{even} - p_{odd}$, because paths that fall an even number of times into $S_1$ contribute 1, while the other paths contribute $-1$. Then balance the $\frac{1}{|V|}$ factor between the two all one vectors, giving each a $\frac{1}{\sqrt{|V|}}$ coefficient and turning them into the normalized vector $\mathbf{1}$.

**Item 3.** Using the distributive law,

$$\begin{aligned} p_{even}(S_1) - p_{odd}(S_1) &= \sum_{b_0,\dots,b_t \in \{0,1\}} (-1)^{b_0+\dots+b_t} \mathbf{1}^\dagger \Pi_{b_t} G \Pi_{b_{t-1}} \dots \Pi_{b_2} G \Pi_{b_1} G \Pi_{b_0} \mathbf{1} \\ &= \mathbf{1}^\dagger (\sum_{b_t \in \{0,1\}} (-1)^{b_t} \Pi_{b_t}) G \dots (\sum_{b_1 \in \{0,1\}} (-1)^{b_1} \Pi_{b_1}) G (\sum_{b_0 \in \{0,1\}} (-1)^{b_0} \Pi_{b_0}) \mathbf{1} \\ &= \mathbf{1}^\dagger (\Pi G)^t \Pi \mathbf{1}. \end{aligned}$$

**Item 4.** Let $v$ be a norm 1 vector. We represent $v = v^{\|} + v^{\perp}$. Notice that $Gv^{\|} = v^{\|} = \left\|v^{\|}\right\|\mathbf{1}$. Therefore,

$$\begin{aligned}
\|\Pi G \Pi G v\| &\leq \left\|\Pi G \Pi G v^{\|}\right\| + \left\|\Pi G \Pi G v^{\perp}\right\| \\
&\leq \left\|v^{\|}\right\| \|\Pi G \Pi \mathbf{1}\| + \|\Pi G \Pi\| \left\|G v^{\perp}\right\| \\
&\leq \|\Pi G \Pi \mathbf{1}\| + \left\|G v^{\perp}\right\| \\
&\leq \left\|\Pi G (\Pi \mathbf{1})^{\|}\right\| + \left\|\Pi G (\Pi \mathbf{1})^{\perp}\right\| + \left\|G v^{\perp}\right\| \\
&\leq \left\|(\Pi \mathbf{1})^{\|}\right\| + 2\lambda.
\end{aligned}$$

To finish the proof note that $\left\|(\Pi\mathbf{1})^{\|}\right\| = |\langle \Pi\mathbf{1}, \mathbf{1}\rangle| = |\frac{|S_0|-|S_1|}{n'}|$. How large can $|\frac{|S_0|-|S_1|}{n'}|$ be? If we define

$$S_{n,b} = \{v \in [n] \mid \langle Z(v), \alpha\rangle = b\},$$

then, as $\Upsilon_0$ is $\varepsilon_0$–biased we know that $|\frac{|S_{n,0}|}{n} - \frac{|S_{n,1}|}{n}| \leq \varepsilon_0$. We deleted $n - n' \leq \beta n$ elements and $|\,|S_{n',0}| - |S_{n',1}|\,|$ is maximized if all the elements in $[n] \setminus [n']$ belong to the smaller set. Thus,

$$|\,|S_0| - |S_1|\,| \quad \leq \quad \frac{1+\varepsilon_0}{2}n - (\frac{1-\varepsilon_0}{2}n - \beta n) \leq (\varepsilon_0 + \beta)n \leq (\varepsilon_0+\beta)\frac{1}{1-\beta}n' \leq (\varepsilon_0 + 2\beta)n'.$$

It therefore follows that $\left\|(\Pi\mathbf{1})^{\|}\right\| \leq \varepsilon_0 + 2\beta$.

**Item 5.**

$$\begin{aligned}
Bias(\Upsilon) \;&=\; Bias_{\alpha}(\Upsilon) \;=\; |p_{even}(S_1) - p_{odd}(S_{n1})| \;=\; |\mathbf{1}^{\dagger}(\Pi G)^t \Pi \mathbf{1}| \\
&\leq\; \left\|(\Pi G)^t\right\| \leq \left\|(\Pi G)^2\right\|^{\lfloor t/2 \rfloor} \;\leq\; (\varepsilon_0 + 2\beta + 2\lambda)^{\lfloor t/2 \rfloor}.
\end{aligned}$$

■

Roughly speaking, Theorem 15(4) tells us that every two steps on the graph reduce the error by $\lambda$, i.e., one in every two steps works for us. For example, if we start with a vector in $\mathcal{V}^{\|}$ and apply $\Pi G \Pi G$ on it, then the first application of $G$ is wasted and then $\Pi$ sends most of the weight to $\mathcal{V}^{\perp}$, because the part that is sent to $\mathcal{V}^{\|}$ is proportional to the bias and is small. The second application of $G$ shrinks the vector by $\lambda$, and we have made two steps one of which worked for us. The first step $\Pi G$ alone did not make any progress, and the algebraic manifestation of this is that $\|\Pi G\| = 1$ because $\|\Pi G \mathbf{1}\| = \|\Pi \mathbf{1}\| = 1$ ($\Pi\mathbf{1}$ in the standard basis $\{\overrightarrow{v} \mid v \in V_1\}$ has $\pm\frac{1}{\sqrt{|V|}}$ entries).

One could wonder whether this bad scenario can be sustained in a long walk of length $t$, and whether, perhaps, most of the steps in a $t$-long walk work for us. While we do not know the answer to this we suspect the answer is negative. If we think of the previous scenario, we see that a bad situation can happen if the second application of $\Pi$ sends most of the vector $G\Pi G\mathbf{1}$ to $V^{\|}$. If this happens, we will keep iterating the bad scenario from before, and only one in two steps will work for us. We think this situation might be possible because $\Pi$ is a global operator, mingling the whole space $\mathcal{V}$. In a sense, we obtain the better solution by "protecting" the parallel space from the action of $\Pi$ and we will see that in detail later on.

Next, we show explicitness:

**Lemma 16.** *If $G$ and $\Upsilon_0$ are explicit then $\Upsilon$ is explicit. If $G$ and $\Upsilon_0$ are fully explicit then $\Upsilon$ is fully explicit.*

**Proof:** $\Upsilon$ is an $\varepsilon$–biased distribution over $\{0,1\}^k$ with support size $n'D^t$. The input to the indexing algorithm is $k, \varepsilon, a \in [n'D^t], j \in [k]$. Given $k$ and $\varepsilon$ the algorithm determines the parameters $n, n', D, t$, and $\varepsilon_0$. The algorithm then interprets $a$ as $a = (u, (i_0, \ldots, i_{t-1}))$ with $u \in V = [n']$ and $i_j \in [D]$. Next, the algorithm computes $PATH(u, a)$. This can be done in time $t \cdot \text{poly}(\log n)$ if $G$ is fully explicit and in time $t \cdot poly(n)$ if $G$ is explicit.

Say $PATH(u, a) = (v_0, \ldots, v_t)$. For each $1 \le i \le t$ the algorithm computes the $j$'th bit of the $v_i$'th element in the support of $\Upsilon_0$. This can be done in time $\text{poly}(\log n)$ if $\Upsilon_0$ is fully explicit, and in time $poly(n)$ if $\Upsilon_0$ is explicit. The output is the sum (mod 2) of all these bits. ■

Altogether,

**Proof:** (of Theorem 8) We use the construction above. By Theorem 15 the construction is $\varepsilon$–biased. The support size is $n'D^t \le nD^t = O(k) \cdot D^{O(\log \frac{1}{\varepsilon})} = \frac{n}{\varepsilon^{O(1)}}$. The construction is explicit because of Lemma 16, and because $G$ is explicit by Lemma12 and $\Upsilon_0$ is fully explicit by Lemma 7. ■

We note the support size can be brought to $\frac{k}{\varepsilon^{4+\alpha}}$ for any constant $\alpha > 0$. However, this is not essential to us, so we do not try to optimize parameters here.

## 4 The more efficient error reduction

### 4.1 Reducing the error with an $s$-wide walk

Suppose

- $\Upsilon_0$ is $\varepsilon_0$–biased over $\{0,1\}^k$ with support size $n$,
- $G$ is $(N_1 = n', D_1, \lambda_1)$ expander with a local inversion function $\phi$, $n' \le n$, $n - n' \le \beta n \le 2\beta n'$.
- $H$ is $(N_2 = D_1^s, D_2, \lambda_2)$ expander.

Define a new distribution $\Upsilon$ over $\{0,1\}^k$ obtained as follows:

1. Sample $v \in [N_1] \times [N_2]$ and $\bar{i} = (i_0, \ldots, i_{t-1}) \in [D_2]^t$, and set $(v_0, \ldots, v_t) = PATH(v, \bar{i})$.
2. Output $Z(v_0^{(1)}) \oplus \ldots \oplus Z(v_t^{(1)}) \in \{0,1\}^k$.

The heart of the construction is proving:

**Theorem 17.** *If $H$ is $\zeta$-pseudorandom with respect to $\phi$, where we define pseudo-randomness with respect to $\phi$ in Definition 25, and,*

$$\varepsilon_0 + 2\beta + 2\lambda_1 \quad \le \quad \lambda_2^2, \tag{3}$$

*then, $\Upsilon$ is $\varepsilon = (\lambda_2^s + s\lambda_2^{s-1} + s^2(\lambda_2^{s-3} + \zeta))^{\lfloor t/s \rfloor}$-biased.*

## 4.2 A top-down view of the proof

We now give a top-down overview of the proof. Notice that since for the time being we are only concerned with the bias achieved by $\Upsilon$ and not with the support size, efficiency, or even the mere existence of $G$,$H$ and $\Upsilon_0$, we can almost completely ignore the parameters, with Eq (3) being the only exception.

We want to express the $s$-wide walk as a composition of linear operators. For $i \in \{1,2\}$, we define a vector space $\mathcal{V}_i$ with $\dim(\mathcal{V}_i) = |V_i| = N_i$, and we identify an element $v^{(i)} \in V_i$ with a basis vector $\overrightarrow{v^{(i)}} \in \mathcal{V}_i$. Notice that $\left\{\overrightarrow{v^{(1)}} \otimes \overrightarrow{v^{(2)}} \mid v^{(1)} \in V_1, v^{(2)} \in V_2\right\}$ is a basis for $\mathcal{V} = \mathcal{V}_1 \otimes \mathcal{V}_2$. On this basis we define the linear operators

$$\begin{aligned} \tilde{H}\left(\overrightarrow{v^{(1)}} \otimes \overrightarrow{v^{(2)}}\right) &= \overrightarrow{v^{(1)}} \otimes \overrightarrow{Hv^{(2)}}, \\ \dot{G}_\ell\left(\overrightarrow{v^{(1)}} \otimes \overrightarrow{v^{(2)}}\right) &= \overrightarrow{v^{(1)}[\pi_{\ell \bmod s}(v^{(2)})]} \otimes \overrightarrow{\psi_{\ell \bmod s}(v^{(2)})}, \end{aligned}$$

where $\psi_{\ell \bmod s}$ is as defined in Equation (2).

Next, fix the linear test $\alpha$ maximizing the bias of $\Upsilon$ and as before let $S_0$ and $S_1$ be the $\varepsilon_0$–balanced partition of $[n]$ associated with it, $S_b = \left\{v \in V^{(1)} \mid \langle Z(v)|\alpha\rangle = b\right\}$. We let $\Pi_b$ be the projection on the vector space $Span(\{\overrightarrow{v} \mid v \in S_b\})$ and $\Pi = \Pi_0 - \Pi_1$. Finally, we extend these operators to $\mathcal{V} = \mathcal{V}_1 \otimes \mathcal{V}_2$ and define

$$\begin{aligned} \dot{\Pi}_b\left(\overrightarrow{v^{(1)}} \otimes \overrightarrow{v^{(2)}}\right) &= \overrightarrow{\Pi_b(v^{(1)})} \otimes \overrightarrow{v^{(2)}} \\ \dot{\Pi}\left(\overrightarrow{v^{(1)}} \otimes \overrightarrow{v^{(2)}}\right) &= \overrightarrow{\Pi(v^{(1)})} \otimes \overrightarrow{v^{(2)}} = \overrightarrow{\Pi_0(v^{(1)})} \otimes \overrightarrow{v^{(2)}} - \overrightarrow{\Pi_1(v^{(1)})} \otimes \overrightarrow{v^{(2)}}. \end{aligned}$$

Let $p_{even}(S_1)$ be the probability a random walk of length $t$ on $G$ visits $S_1 \subseteq [n']$ an even number of times. Similar to Section 3,

**Theorem 18.** *Denote $\dot{L}_j = \dot{\Pi}\dot{G}_j\tilde{H}$. We have:*

1. $Bias(\Upsilon) = Bias_\alpha(\Upsilon) = |p_{even}(S_1) - p_{odd}(S_1)|$,
2. $p_{even}(S_1) - p_{odd}(S_1) = \sum_{b_0,\ldots,b_t \in \{0,1\}} (-1)^{b_0+\ldots+b_t} \mathbf{1}\dot{\Pi}_{b_t}\dot{G}_{t-1}\tilde{H}\dot{\Pi}_{b_{t-2}} \ldots \dot{\Pi}_{b_2}\dot{G}_1\tilde{H}\dot{\Pi}_{b_1}\dot{G}_0\tilde{H}\dot{\Pi}_{b_0}\mathbf{1}$,
3. $p_{even}(S_1) - p_{odd}(S_1) = \mathbf{1}\dot{\Pi}\dot{G}_{t-1}\tilde{H}\dot{\Pi} \ldots \dot{\Pi}\dot{G}_1\tilde{H}\dot{\Pi}\dot{G}_0\tilde{H}\dot{\Pi}\mathbf{1}$,
4. $Bias(\Upsilon) \leq \left\|\dot{L}_{s-1} \ldots \dot{L}_0\right\|^{\lfloor t/s \rfloor}$.
5. *If $H$ is $\zeta$-pseudorandom with respect to $\phi$ and $\varepsilon_0 + 2\beta + 2\lambda_1 \leq \lambda_2^2$ then* $\left\|\dot{L}_{s-1} \ldots \dot{L}_0\right\| \leq \lambda_2^s + s\lambda_2^{s-1} + s^2(\lambda_2^{s-3} + \zeta)$.

**Proof:**

**Item 1.** The proof is identical to the proof of Theorem 15(1).

**Item 2.** $\frac{1}{|V|}\begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix}$ corresponds the uniform distribution over $V=[N_1]\times[N_2]$.

$$\begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix}\dot{\Pi}_{b_t}\dot{G}_{t-1}\dots\tilde{H}\dot{\Pi}_{b_0}\frac{1}{|V|}\begin{pmatrix}1\\ 1\\ \vdots\\ 1\end{pmatrix}$$

is the probability that $v_j \in S_{b_j}$ for $j=0,\dots t$. The rest is almost identical to Theorem 15(2).

**Item 3.** Using the distributive law as in Theorem 15(3).

**Item 4.**

$$\begin{aligned} Bias(\Upsilon) &= Bias_\alpha(\Upsilon) \;=\; |p_{even}(S_1)-p_{odd}(S_1)| \\ &= |\mathbf{1}\dot{L}_{t-1}\dots\dot{L}_0\dot{\Pi}\mathbf{1}| \le \left\|\dot{L}_{t-1}\dots\dot{L}_0\dot{\Pi}\right\| \le \left\|\dot{L}_{t-1}\dots\dot{L}_0\right\|. \end{aligned}$$

However, $\dot{L}_i=\dot{\Pi}\dot{G}_i\tilde{H}$ and $\dot{G}_i=\dot{G}_{i \bmod s}$. Hence $\dot{L}_{t-1}\dots\dot{L}_0$ has $\dot{L}_{s-1}\dots\dot{L}_0$ repeated $\lfloor\frac{t}{s}\rfloor$ times, plus some trailing operators, and, $Bias(\Upsilon)\le\left\|\dot{L}_{t-1}\dots\dot{L}_0\right\|\le\left\|\dot{L}_{s-1}\dots\dot{L}_0\right\|^{\lfloor t/s\rfloor}$.

**Item 5.** We define pseudo-randomness with respect do $\phi$ in Def 25 and prove (5) in Theorem 24 in Section 6.3. This is the main technical work in the paper.

■

Items (4) and (5) together give Theorem 17.

## 5 The construction with parameters

### 5.1 Choosing parameters

We are given $k,\varepsilon>0$. We fix a parameter $\alpha\le\frac{1}{100}$ such that

$$\frac{\alpha^3}{2\log\frac{1}{\alpha}} \;\ge\; \frac{1}{\log\frac{1}{\varepsilon}}. \tag{4}$$

Our goal is to build an $\varepsilon$–biased distribution over $\{0,1\}^k$ with support size $n=O(\frac{k}{\varepsilon^{2(1+14\alpha)}})$.

We may assume that $\frac{1}{100}$ respects Eq (4), for otherwise $\varepsilon$ is also a constant and we may use Theorem 8. If we take the smallest possible value of $\alpha$ allowed by Eq (4) then $\alpha=\Theta((\frac{\log\log\frac{1}{\varepsilon}}{\log\frac{1}{\varepsilon}})^{1/3})$. For a first reading it is recommended to think of $\alpha$ as an arbitrarily small constant. We also remark that w.l.o.g. we can assume that $(\frac{1}{\varepsilon})^{8\alpha}<k$, for otherwise the AGHP construction [3] has support size $O(\frac{k^2}{\varepsilon^2})=O(\frac{k}{\varepsilon^{2+8\alpha}})$ and we are done.

We now choose parameters for:

- An inner graph: An $(N_2, D_2, \lambda_2)$ expander graph $H(V_2, E_2)$.
- A Large-Bias distribution: An $\varepsilon_0$–biased distribution $\Upsilon_0$ over $\{0,1\}^k$ with support size $n$.
- An outer graph: An $(N_1, D_1, \lambda_1)$ expander graph $G(V_1, E_1)$ with local inversion function $\phi : [D_1] \to [D_1]$.

We choose the parameters as follows:

**The inner graph** $H$ : For reasons that will become clear later, the inner graph is a Cayley graph over the Abelian group $\mathbb{Z}_2^{\log |V_2|}$. We choose it as follows:

- Set $s = \lceil \frac{1}{\alpha} \rceil$ and let $D_2$ be the first power of 2 larger than $s^{4s}$.
- Define $\lambda_2 = \frac{b_2}{\sqrt{D_2}}$ for $b_2 = 4\sqrt{2}\log(D_2)s$. We remark that $D_2 \geq (2b_2 s^2)^2$ because $\alpha \leq \frac{1}{100}$.
- Let $A$ be an explicit $\lambda_2$–biased set over $\{0,1\}^{4s\log(D_2)}$ with support size $2^{\frac{(4s\log(D_2))^2}{\lambda_2^2}} = (\frac{b_2}{\lambda_2})^2$ using the AGHP construction (see Lemma 6). $N_2 = 2^{4s\log(D_2)} = D_2^{4s}$.
- Let $H = Cay(\mathbb{Z}_{4s\log(D_2)}, A)$.
  By Lemma 11 $H$ is a $(N_2 = 2^{4s\log(D_2)} = D_2^{4s}, (\frac{b_2}{\lambda_2})^2 = D_2, \lambda_2)$ graph.
  By Lemma 29 in Section 7 that we still have to prove, $H$ is $\zeta = 0$–pseudorandom with respect to any locally invertible function $\phi : [D_1] \to [D_1]$.

**The distribution** $\Upsilon_0$ : Set $\varepsilon_0 = \frac{1}{D_2}$. Pick an explicit $\varepsilon_0$–biased distribution $\Upsilon_0$ over $\{0,1\}^k$ with support size $n = O(\frac{k}{\varepsilon_0^c})$ using Theorem 8.[5] Similarly, we can construct such a distribution with support size $2n$, e.g., by taking each element in the support twice.

**The Outer graph** $G$ : Set $D_1 = D_2^4$. Use Lemma 12 with $n, D$ and $\beta = \frac{\lambda_2^2}{6}$ to find an explicit undirected graph $G$ that is a $(n', D_1, \lambda_1)$ Ramanujan expander with $D_1 \leq \frac{8}{\lambda_1^2}$. The number of vertices $n'$ is either close to $n$ or to $2n$. W.l.o.g. it is close to $n$ (otherwise take every element in the support twice), and $\frac{n-n'}{n'} \leq \beta n \leq 2\beta n'$. We notice a few things:

- $\lambda_1 \leq \frac{\lambda_2^2}{6}$. This is because $\lambda_1 \leq \frac{4}{\sqrt{D_1}} = \frac{4}{D_2^2} \leq \frac{b_2^2}{6D_2} = \frac{\lambda_2^2}{6}$. Also, $\varepsilon_0 = \frac{1}{D_2} \leq \frac{b_2^2}{3D_2} = \frac{\lambda_2^2}{3}$. Together we see that $\varepsilon_0 + 2\beta + 2\lambda_1 \leq \lambda_2^2$.
- Also, $N_2 = D_2^{4s} = D_1^s$.
- We also remark that $N_2 = D_2^{4s} = s^{16s^2} = (2^{\log\frac{1}{\alpha}})^{\frac{16}{\alpha^2}} = 2^{16\alpha\frac{\log\frac{1}{\alpha}}{\alpha^3}} \leq 2^{16\alpha\frac{\log\frac{1}{\varepsilon}}{2}} \leq (\frac{1}{\varepsilon})^{8\alpha}$. We explained at the beginning of this section that we can assume w.l.o.g. that $(\frac{1}{\varepsilon})^{8\alpha} \leq k$. Hence, we can assume $N_2 \leq k$.

**The walk length** : Set the walk length $t$ to be the first integer such that $\lambda_2^{(1-4\alpha)(1-\alpha)t} \leq \varepsilon$.

Notice that

[5]We can also work with other constant-biased distributions, and in particular with Justensen amplified with a true product a constant number of times to reach the desired small bias.

$$\begin{aligned}(\frac{1}{\lambda_2})^{(1-4\alpha)(1-\alpha)\frac{s}{\alpha}} &< (\frac{1}{\lambda_2})^{s/\alpha} = (\frac{\sqrt{D_2}}{b_2})^{s/\alpha} \le D_2^{s/2\alpha} \\ &= (2^{4s\log s})^{s/2\alpha} = 2^{2\frac{\log\frac{1}{\alpha}}{\alpha^3}} \le 2^{\log\frac{1}{\varepsilon}} = \frac{1}{\varepsilon},\end{aligned}$$

because $D_2$ is about $s^{4s} = 2^{4s\log(s)}$, $s$ is about $\frac{1}{\alpha}$ and because of our choice of $\alpha$. Hence $\lambda_2^{(1-4\alpha)(1-\alpha)\frac{s}{\alpha}} > \varepsilon$. It follows that $t \ge \frac{s}{\alpha}$.

## 5.2 Bias

**Theorem 19.** *$H$ is $\zeta = 0$ pseudo-random with respect to $\phi$.*

We will prove the theorem in Section 7. With that:

**Theorem 20.** *$\Upsilon$ is $\varepsilon$– biased.*

**Proof:** We have seen in Theorem 17 that $\Upsilon$ is $(\lambda_2^s + s\lambda_2^{s-1} + s^2\lambda_2^{s-3})^{\lfloor t/s\rfloor}$-biased. However,

$$\begin{aligned}(\lambda_2^s + s\lambda_2^{s-1} + s^2\lambda_2^{s-3})^{\lfloor t/s\rfloor} &\le (2s^2\lambda_2^{s-3})^{\lfloor t/s\rfloor} \le (\lambda_2^{s-4})^{\frac{t}{s}-1} \\ &= \lambda_2^{\frac{s-4}{s}\cdot(t-s)} = \lambda_2^{(1-\frac{4}{s})(1-\frac{s}{t})t} \le \lambda_2^{(1-4\alpha)(1-\alpha)t} = \varepsilon,\end{aligned}$$

where we have used $\lambda_2 = \frac{b_2}{\sqrt{D_2}} \le \frac{1}{2s^2}$, $s \ge \frac{1}{\alpha}$ and $t \ge \frac{s}{\alpha}$. ■

## 5.3 Support size

**Theorem 21.** *$\Upsilon$ has support size $O(\frac{k}{\varepsilon^{2+O(\alpha)}})$.*

**Proof:** The support size is $N_1N_2D_2^t = n'D_1^sD_2^t < nD_1^sD_2^t$ and

$$\begin{aligned}nD_1^sD_2^t &= \Theta(\frac{k}{\varepsilon_0^c}D_2^{4s}D_2^t) = \Theta(kD_2^{t+4s+c}) \\ &= O(kD_2^{t+4\alpha t+c}) = O(kD_2^{(1+5\alpha)t}).\end{aligned}$$

However, $D_2 \ge (s^4)^s \ge b_2^s = b_2^{1/\alpha}$, because $b_2 = 4\sqrt{2}\log(D_2)s \le s^4$. It therefore follows that $D_2^{\frac{1}{2}-\alpha} = \frac{\sqrt{D_2}}{D_2^\alpha} \le \frac{\sqrt{D_2}}{b_2} = \frac{1}{\lambda_2}$ and $D_2 \le (\frac{1}{\lambda_2})^{\frac{1}{\frac{1}{2}-\alpha}}$. Hence,

$$\begin{aligned}D_2^t &\le (\frac{1}{\lambda_2})^{\frac{t}{\frac{1}{2}-\alpha}} = (\frac{1}{\lambda_2})^{\frac{2t}{1-2\alpha}} = (\frac{1}{\varepsilon})^{\frac{1}{(1-4\alpha)(1-\alpha)t}\frac{2t}{1-2\alpha}} \le (\frac{1}{\varepsilon})^{2(1+8\alpha)}, \text{ and,} \\ D_2^{(1+5\alpha)t} &\le (\frac{1}{\varepsilon})^{2(1+8\alpha)(1+5\alpha)} \le (\frac{1}{\varepsilon})^{2(1+14\alpha)},\end{aligned}$$

where the last inequalities hold for $\alpha$ small enough. Altogether the support size is $O(\frac{k}{\varepsilon^{2(1+O(\alpha))}})$ as desired!
■

We now plug in a value for $\alpha$. Plugging $\alpha$ a small enough constant gives a construction with support size $\frac{k}{\varepsilon^{2+\beta}}$ for arbitrarily small constant $\beta$. plugging the smallest value of $\alpha$ allowed by Eq (4) we get:

**Lemma 22.** *If we take $\alpha$ having equality in Eq (4), then the construction has support size*

$$k \cdot 2^{2\log\frac{1}{\varepsilon}+O((\log\frac{1}{\varepsilon})^{2/3}(\log\log\frac{1}{\varepsilon})^{1/3})}.$$

**Proof:** Take such an $\alpha$. Then $\log\frac{1}{\alpha} = \Theta(\log\log\frac{1}{\varepsilon})$ and

$$\alpha \;=\; \Theta((\frac{\log\frac{1}{\alpha}}{\log\frac{1}{\varepsilon}})^{1/3}).$$

Hence,

$$(\frac{1}{\varepsilon})^{O(\alpha)} \;=\; 2^{O(\log\frac{1}{\varepsilon}\cdot(\frac{\log\log\frac{1}{\varepsilon}}{\log\frac{1}{\varepsilon}})^{1/3})} \;=\; 2^{O((\log\frac{1}{\varepsilon})^{2/3}(\log\log\frac{1}{\varepsilon})^{1/3})}.$$

■

In the introduction we give an intuitive explanation of the parameters and the choice of $\alpha$.

## 5.4 Explicitness

**Claim 23.** *If $A, G$ and $\Upsilon_0$ are explicit then $\Upsilon$ is explicit. If $A, G$ and $\Upsilon_0$ are fully explicit then $\Upsilon$ is fully explicit.*

**Proof:** $\Upsilon$ is a distribution over $\{0,1\}^k$ with support size $N_1 N_2 D_2^t$. We need to show an efficient indexing algorithm for $\Upsilon$. The input to the indexing algorithm is $k, \varepsilon, a \in [N_1 N_2 D_2^t], j \in [k]$. Given $k$ and $\varepsilon$ the algorithm computes $\alpha$ and then determines all the parameters given above including: $N_2, D_2$, $N_1, D_1$, $n$ and $\varepsilon_0$. All these parameters take at most $O(\log\frac{k}{\varepsilon})$ bits to represent, or approximate accurately enough.

The algorithm then interprets $a$ as $a = (u^{(1)}, u^{(2)}, (i_0, \ldots, i_{t-1}))$ with $u^{(1)} \in V_1, u^{(2)} \in V_2$ and $i_j \in [D_2]$. We remind the reader that $|V_1| = \text{poly}(k/\varepsilon)$ and w.l.o.g we can assume $N_2 \le k$. Next, the algorithm uses the graphs $H = Cay(\mathbb{Z}_2^{4s\log(D_2)}, A)$ and $G$. The algorithm computes $PATH((u^{(1)}, u^{(2)}), (i_0, \ldots, i_{t-1}))$. This can be done in time $t \cdot \text{poly}(n)$ if $G$ and $H$ are explicit and time $t \cdot \text{poly}(\log n)$ if $G$ and $H$ are fully explicit.

Say $PATH((u^{(1)}, u^{(2)}), (i_0, \ldots, i_{t-1})) = (v_0, \ldots, v_t)$. The algorithm defines the $\varepsilon_0$ biased distribution $\Upsilon_0$ and for each $1 \le i \le t$ computes the $j$'th bit of the $v_i$'th element in the support of $\Upsilon_0$. This can be done in time $poly(n)$ if $\Upsilon_0$ is explicit, and in time $\text{poly}(\log k)$ if $\Upsilon_0$ is fully explicit. The output is the sum (mod 2) of all these bits. ■

Since $A, G$ and $\Upsilon_0$ are explicit (by Lemma 6, Lemma 12 and Theorem 8 respectively) we conclude that $\Upsilon$ is explicit.

### 5.5 What have we omitted so far?

Thus, in order to complete the proof we still need to:

- Define pseudo-randomness with respect to $\phi$. We do that in Definition 25.
- Prove Theorem 24 that bounds $\|L_{s-1}\dots L_0\|$ when $H$ is pseudo-random with respect to $\phi$. We do that in Section 6.
- Prove Theorem 19 that $H$ is pseudo-random with respect to $\phi$. We do in Section 7.
- We need to show that the basic structures that we use are explicit, namely prove Lemmas 6, 7 and 12. We do that in Section 8.

## 6 Understanding $L_{s-1}\dots L_0$

In this section we prove:

**Theorem 24.** *If $H$ is $\zeta$-pseudorandom with respect to $\phi$ and $\varepsilon_0+2\beta+2\lambda_1\le\lambda_2^2$ then $\left\|\dot{L}_{s-1}\dots\dot{L}_0\right\|\le\lambda_2^s+s\lambda_2^{s-1}+s^2(\lambda_2^{s-3}+\zeta)$.*

### 6.1 The action of $L_i\cdot\ldots\cdot L_0$ on product vectors

In this subsection we would like to understand the action of $\dot{L}_i=\dot{\Pi}\dot{G}_i\tilde{H}$ on a product vector $u^{(1)}\otimes u^{(2)}$ for $0\le i\le s-1$. We remind the reader that:

- $G$ is $D_1$ regular and can be represented as $G=\frac{1}{D_1}\sum_{i=1}^{D_1}\mathcal{G}_i$ where $\mathcal{G}_i$ is the transition matrix of some permutation in $\mathbb{S}_{V_1}$. Also, $G$ has a local inversion function $\phi$.
- $\dot{G}_i\left(\overrightarrow{v^{(1)}}\otimes\overrightarrow{v^{(2)}}\right)=\overrightarrow{\mathcal{G}_{\pi_i(v^{(2)})}(v^{(1)})}\otimes\overrightarrow{\psi_i(v^{(2)})}$.
- $\dot{\Pi}=\Pi\otimes I_{\mathcal{V}_2}$,
- $H$ is $D_2$ regular and can be represented as $H=\frac{1}{D_2}\sum_{j=1}^{D_2}\mathcal{H}_j$ where $\mathcal{H}_j$ is the transition matrix of a permutation $\gamma_j\in\mathbb{S}_{V_2}$. Let $\Gamma_j$ be the linear operator on $\mathcal{V}_2$ corresponding to the permutation $\gamma_j$ and $\tilde{\Gamma}_j=I_{\mathcal{V}_1}\otimes\Gamma_j$.
- $\tilde{H}=I_{\mathcal{V}_1}\otimes H=\frac{1}{D_2}\sum_{j=1}^{D_2}I_{\mathcal{V}_1}\otimes\Gamma_j=\sum_{j=1}^{D_2}\tilde{\Gamma}_j$ and therefore $\dot{L}_i=\dot{\Pi}\dot{G}_i\tilde{H}=\frac{1}{D_2}\sum_{j=1}^{D_2}\dot{\Pi}\dot{G}_i\tilde{\Gamma}_j$.

We therefore focus our attention on understanding the action of $\dot{\Pi}\dot{G}_i\tilde{\Gamma}_j$ on $u^{(1)}\otimes u^{(2)}$. We have:

$$\begin{aligned}\dot{\Pi}\dot{G}_i\tilde{\Gamma}_j\;(\overrightarrow{u^{(1)}}\otimes\overrightarrow{u^{(2)}}) &= \dot{\Pi}\dot{G}_i(\overrightarrow{u^{(1)}}\otimes\overrightarrow{\gamma_j(u^{(2)})})\\ &= \dot{\Pi}(\overrightarrow{\mathcal{G}_{\pi_i(\gamma_j(u^{(2)}))}(u^{(1)})}\otimes\overrightarrow{\psi_i(\gamma_j(u^{(2)}))})\\ &= \overrightarrow{\Pi(\mathcal{G}_{\pi_i(\gamma_j(u^{(2)}))}(u^{(1)}))}\otimes\overrightarrow{\psi_i(\gamma_j(u^{(2)}))}\end{aligned}$$

The crucial point is that $\dot{\Pi G_i}\tilde{\Gamma}_j$ on a product vector $u^{(1)} \otimes u^{(2)}$ is a product vector. Now, $L_i \dots L_0$ is a convex (and uniform) combination of $\dot{\Pi G_i}\tilde{\Gamma}_{j_i} \dots \dot{\Pi G_0}\tilde{\Gamma}_{j_0}$ for all the $D_2^{i+1}$ sequences $(j_0, \dots, j_i) \in [D_2]^{i+1}$, and,

$$\dot{\Pi G_i}\tilde{\Gamma}_{j_i} \dots \dot{\Pi G_0}\tilde{\Gamma}_{j_0}(\overrightarrow{u^{(1)}} \otimes \overrightarrow{u^{(2)}}) \quad = \quad \overrightarrow{\Pi \mathcal{G}_{\pi_i(\sigma_{j_0,\dots,j_i}(u^{(2)}))} \dots \Pi \mathcal{G}_{\pi_0(\sigma_{j_0}(u^{(2)}))}(u^{(1)})} \otimes \overrightarrow{\psi_i(\sigma_{j_0,\dots,j_i}(u^{(2)}))} \quad (5)$$

where $\sigma_{j_0}(u^{(2)}) = \gamma_{j_0}(u^{(2)})$ and $\sigma_{j_0,\dots,j_\ell} = \gamma_{j_\ell}(\psi_{\ell-1}(\sigma_{j_0,\dots,j_{\ell-1}}))$ for $\ell \geq 1$. The crucial point is, again, that $\sigma_{j_0,\dots,j_\ell}$ is a composition of permutation on $V_2$, hence it is itself a permutation on $V_2$.

## 6.2 The action of $L_i \cdot \dots \cdot L_0$ on parallel vectors

We keep notations as before.

**Definition 25.** *Let $G$ be an undirected graph with local inversion function $\phi : [D_1] \to [D_1]$. Let $H$ be a $D_2$ regular graph, $H = \frac{1}{D_2}\sum_{j=1}^{D_2} \mathcal{H}_j$ where $\mathcal{H}_j$ is the transition matrix of a permutation $\gamma_j \in \mathbb{S}_{V_2}$. Suppose $\ell \leq s$ and let $\sigma_{j_0,\dots,j_\ell}$ be defined as before, i.e.,*

$$\begin{aligned} \sigma_{j_0}(u^{(2)}) &= \gamma_{j_0}(u^{(2)}), \\ \sigma_{j_0,\dots,j_\ell}(u^{(2)}) &= \gamma_{j_\ell}(\psi_{\ell-1}(\sigma_{j_0,\dots,j_{\ell-1}}(u^{(2)}))). \end{aligned}$$

- *We say $(j_0, \dots, j_{\ell-1})$ is $\zeta$–*pseudorandom *with respect to $\phi$ if:*

  $$\left| \pi_0(\sigma_{j_0}(U)) \circ \dots \circ \pi_{\ell-1}(\sigma_{j_0,\dots,j_{\ell-1}}(U)) - U_{[D_1]^\ell} \right|_1 \quad \leq \quad \zeta,$$

  *where $\pi_0(\sigma_{j_0}(U)) \circ \dots \circ \pi_{\ell-1}(\sigma_{j_0,\dots,j_{\ell-1}}(U))$ is the distribution obtained by picking $v^{(2)} \in V_2$ uniformly at random and outputting $(\pi_0(\sigma_{j_0}(v^{(2)})) \circ \dots \circ \pi_{\ell-1}(\sigma_{j_0,\dots,j_{\ell-1}}(v^{(2)})))$ and $U_{[D_1]^\ell}$ is the uniform distribution over $[D_1]^\ell$.*

- *We say $H$ is $\zeta$–*pseudorandom *with respect to $\phi$ if $(j_0, \dots, j_{s-1})$ is $\zeta$–*pseudorandom *with respect to $\phi$ for every $(j_0, \dots, j_{s-1}) \in [D_2]^s$.*

**Lemma 26.** *Suppose $\ell \leq s$ and $(j_0, \dots, j_{\ell-1})$ is $\zeta$–pseudorandom with respect to $\phi$ and denote by $\tilde{\Gamma}_{j_0}, \dots, \tilde{\Gamma}_{j_{\ell-1}}$ the operators corresponding to $\gamma_{j_0}, \dots, \gamma_{j_{\ell-1}}$. Any $\tau \in \mathcal{V}^{\parallel}$ can be written as $\tau = \tau^{(1)} \otimes \mathbf{1}_{\mathcal{V}_2}$. For any $\tau, \xi \in \mathcal{V}^{\parallel}$:*

$$\left| \left\langle \dot{\Pi G}_{\ell-1}\tilde{\Gamma}_{j_{\ell-1}} \dots \dot{\Pi G}_0\tilde{\Gamma}_{j_0}\tau, \xi \right\rangle - \left\langle (\Pi G)^\ell \tau^{(1)}, \xi^{(1)} \right\rangle \right| \quad \leq \quad \zeta \cdot \|\tau\| \cdot \|\xi\|.$$

**Proof:** By definition $\mathbf{1}_{\mathcal{V}_2} = \frac{1}{\sqrt{N_2}} \sum_{v^{(2)} \in \mathcal{V}_2} v^{(2)}$. Using Eq (5):

$$\begin{aligned} \dot{\Pi G}_{\ell-1}\tilde{\Gamma}_{j_{\ell-1}} \dots \dot{\Pi G}_0\tilde{\Gamma}_{j_0}\tau &= \frac{1}{\sqrt{N_2}} \sum_{v^{(2)} \in V_2} \dot{\Pi G}_{\ell-1}\tilde{\Gamma}_{j_{\ell-1}} \dots \dot{\Pi G}_0\tilde{\Gamma}_{j_0}(\tau^{(1)} \otimes \overrightarrow{v^{(2)}}) \\ &= \frac{1}{\sqrt{N_2}} \sum_{v^{(2)} \in V_2} \Pi \mathcal{G}_{\pi_{\ell-1}(\sigma_{j_0,\dots,j_{\ell-1}}(v^{(2)}))} \dots \Pi \mathcal{G}_{\pi_0(\sigma_{j_0}(v^{(2)}))}(\tau^{(1)}) \otimes \overrightarrow{\sigma(v^{(2)})}, \end{aligned}$$

for some permutation $\sigma : V_2 \to V_2$. Therefore,

$$\begin{aligned}
&\left\langle \dot{\Pi}\dot{G}_{\ell-1}\tilde{\Gamma}_{j_{\ell-1}}\dots\dot{\Pi}\dot{G}_0\tilde{\Gamma}_{j_0}\tau,\xi\right\rangle\\
=\;& \frac{1}{N_2}\sum_{v^{(2)},u^{(2)}\in V_2}\left\langle \Pi\mathcal{G}_{\pi_{\ell-1}(\sigma_{j_0,\dots,j_{\ell-1}}(v^{(2)}))}\dots\Pi\mathcal{G}_{\pi_0(\sigma_{j_0}(v^{(2)}))}(\tau^{(1)})\otimes\overrightarrow{\sigma(v^{(2)})},\xi^{(1)}\otimes\overrightarrow{u^{(2)}}\right\rangle\\
=\;& \frac{1}{N_2}\sum_{v^{(2)},u^{(2)}\in V_2}\left\langle \Pi\mathcal{G}_{\pi_{\ell-1}(\sigma_{j_0,\dots,j_{\ell-1}}(v^{(2)}))}\dots\Pi\mathcal{G}_{\pi_0(\sigma_{j_0}(v^{(2)}))}(\tau^{(1)}),\xi^{(1)}\right\rangle\cdot\left\langle\overrightarrow{\sigma(v^{(2)})},\overrightarrow{u^{(2)}}\right\rangle.
\end{aligned}$$

However, as $\sigma$ is a permutation over $V_2$, for every $v^{(2)}\in V_2$ there is exactly one $u^{(2)}$ that does not vanish. Hence,

$$\begin{aligned}
\left\langle \dot{\Pi}\dot{G}_{\ell-1}\tilde{\Gamma}_{j_{\ell-1}}\dots\dot{\Pi}\dot{G}_0\tilde{\Gamma}_{j_0}\tau,\xi\right\rangle &= \frac{1}{N_2}\sum_{v^{(2)}\in V_2}\left\langle \Pi\mathcal{G}_{\pi_{\ell-1}(\sigma_{j_0,\dots,j_{\ell-1}}(v^{(2)}))}\dots\Pi\mathcal{G}_{\pi_0(\sigma_{j_0}(v^{(2)}))}(\tau^{(1)}),\xi^{(1)}\right\rangle\\
&= \mathbb{E}_{z_0,\dots,z_{\ell-1}\sim\mathcal{Z}}\left\langle \Pi\mathcal{G}_{z_{\ell-1}}\dots\Pi\mathcal{G}_{z_0}\tau^{(1)},\xi^{(1)}\right\rangle,
\end{aligned}$$

where $\mathcal{Z}$ is the distribution on $[D_1]^\ell$ obtained by picking $v^{(2)}$ uniformly at random in $V_2$ and outputting $z_0,\dots,z_{\ell-1}$ where $z_i=\pi_i(\sigma_{j_0,\dots,j_i}(v^{(2)}))$. As $(j_0,\dots,j_{\ell-1})$ is $\zeta$–pseudorandom with respect to $G$ we know that $\left|\mathcal{Z}-U_{[D_1]^\ell}\right|_1\le\zeta$. Also notice that

$$\left\langle(\Pi G)^\ell\tau^{(1)},\xi^{(1)}\right\rangle = \mathbb{E}_{(z_0,\dots,z_{\ell-1})\in[D_1]^\ell}\left\langle \Pi\mathcal{G}_{z_{\ell-1}}\dots\Pi\mathcal{G}_{z_0}\tau^{(1)},\xi^{(1)}\right\rangle$$

By Claim 14,

$$\begin{aligned}
&\left|\left\langle \dot{\Pi}\dot{G}_{\ell-1}\tilde{\Gamma}_{j_{\ell-1}}\dots\dot{\Pi}\dot{G}_0\tilde{\Gamma}_{j_0}\tau,\xi\right\rangle-\left\langle(\Pi G)^\ell\tau^{(1)},\xi^{(1)}\right\rangle\right|\\
=\;& \left|\mathbb{E}_{z\sim\mathcal{Z}}\left\langle \Pi\mathcal{G}_{z_{\ell-1}}\dots\Pi\mathcal{G}_{z_0}\tau^{(1)},\xi^{(1)}\right\rangle-\mathbb{E}_{z\in[D_1]^\ell}\left\langle \Pi\mathcal{G}_{z_{\ell-1}}\dots\Pi\mathcal{G}_{z_0}\tau^{(1)},\xi^{(1)}\right\rangle\right|\\
\le\;& \left|\mathcal{Z}-U_{[D_1]^\ell}\right|_1\cdot\left\|\tau^{(1)}\right\|\cdot\left\|\xi^{(1)}\right\|\le\zeta\cdot\left\|\tau^{(1)}\right\|\cdot\left\|\xi^{(1)}\right\|=\zeta\cdot\|\tau\|\cdot\|\xi\|
\end{aligned}$$

■

We now extend Lemma 26 to $H$ using the fact that $\tilde{H}=\sum_{j=1}^{D_2}\tilde{\Gamma}_j$, .i.e., a convex combination of $\tilde{\Gamma}_j$. We claim:

**Theorem 27.** *Suppose that $H$ is $\zeta$–*pseudorandom *with respect to $\phi$. For every $0\le i_1\le i_2\le s-1$ and every $\tau,\xi\in\mathcal{V}^{\|}$,*

$$\left|\left\langle\dot{L}_{i_1}\dots\dot{L}_{i_2}\tau,\xi\right\rangle-\left\langle(\Pi G)^{i_2-i_1+1}\tau^{(1)},\xi^{(1)}\right\rangle\right| \le \zeta\cdot\|\tau\|\cdot\|\xi\|.$$

**Proof:** Let $\ell=i_2-i_1+1$. As explained before,

$$\dot{L}_{i_1}\dots\dot{L}_{i_2}=\mathbb{E}_{j_1,\dots,j_\ell\in[D_2]}\dot{\Pi}\dot{G}_{i_1}\tilde{\Gamma}_{j_1}\dots\dot{\Pi}\dot{G}_{i_2}\tilde{\Gamma}_{j_\ell}$$

and

$$\left\langle\dot{L}_{i_1}\dots\dot{L}_{i_2}\tau,\xi\right\rangle=\mathbb{E}_{j_1,\dots,j_\ell\in[D_2]}\left\langle\dot{\Pi}\dot{G}_{i_1}\tilde{\Gamma}_{j_1}\dots\dot{\Pi}\dot{G}_{i_2}\tilde{\Gamma}_{j_\ell}\tau,\xi\right\rangle$$

Therefore the theorem follows from Claim 14 that asserts that for every $j_1,\dots,j_\ell$, $\left\langle\dot{\Pi}\dot{G}_{i_1}\tilde{\Gamma}_{j_1}\dots\dot{\Pi}\dot{G}_{i_2}\tilde{\Gamma}_{j_\ell}\tau,\xi\right\rangle$ is $\zeta$ close to $\left\langle(\Pi G)^\ell\tau^{(1)},\xi^{(1)}\right\rangle$. ■

## 6.3 Bounding $\left\|\dot{L}_{s-1}\dots\dot{L}_0\right\|$

We now prove Theorem 24 that states that if $H$ is $\zeta$-pseudorandom with respect to $\phi$ and $\varepsilon_0+2\beta+2\lambda_1 \le \lambda_2^2$ then $\left\|\dot{L}_{s-1}\dots\dot{L}_0\right\| \le \lambda_2^s + s\lambda_2^{s-1} + s^2(\lambda_2^{s-3}+\zeta)$.

**Proof:** Let $w_s, x_0$ be norm 1 vectors such that

$$\left\|\dot{L}_{s-1}\dots\dot{L}_0\right\| \;=\; w_s^\dagger \dot{L}_{s-1}\dots\dot{L}_0 x_0. \tag{6}$$

We would like to decompose the expression (6) until the vectors on both sides are parallel. For $i \ge 1$ define

$$\begin{aligned} x_i &= \dot{L}_{i-1}x_{i-1}^\perp \;=\; \dot{\Pi}\dot{G}_{i-1}\tilde{H}x_{i-1}^\perp, \\ w_{s-i} &= \tilde{H}(\dot{G}_{s-i}\dot{\Pi}w_{s-i+1})^\perp, \\ p_{s-i} &= \tilde{H}(\dot{G}_{s-i}\dot{\Pi}w_{s-i+1})^\parallel \end{aligned}$$

Recall that $\left\|\tilde{H}v\right\| \le \lambda_2\|v\|$ for $v \in \mathcal{V}^\perp$, and $\tilde{H}v = v$ for $v \in \mathcal{V}^\parallel$. Also, for $i < s$, $w_i \in \mathcal{V}^\perp$ and $p_i \in \mathcal{V}^\parallel$. Thus, for $i > 0$,

$$\begin{aligned} \|x_i\| &= \left\|\dot{L}_{i-1}x_{i-1}^\perp\right\| = \left\|\dot{\Pi}\dot{G}_{i-1}\tilde{H}x_{i-1}^\perp\right\| \le \left\|\tilde{H}x_{i-1}^\perp\right\| \le \lambda_2\left\|x_{i-1}^\perp\right\| \le \lambda_2\|x_{i-1}\|, \\ \|w_{s-i}\| &= \left\|\tilde{H}(\dot{G}_{s-i}\dot{\Pi}w_{s-i+1})^\perp\right\| \le \lambda_2\|w_{s-i+1}\|, \\ \|p_{s-i}\| &= \left\|\tilde{H}(\dot{G}_{s-i}\dot{\Pi}w_{s-i+1})^\parallel\right\| \le \|w_{s-i+1}\|. \end{aligned}$$

Therefore $\|x_i\| \le \lambda_2^i$, $\|w_i\| \le \lambda_2^{s-i}$ and $\|p_i\| \le \lambda_2^{s-i-1}$. Decomposing (6) we get:

**Lemma 28.**

$$w_s^\dagger \dot{L}_{s-1}\dots\dot{L}_0 x_0 \;=\; w_s^\dagger x_s^\perp + \sum_{s\ge j\ge i\ge 0} (w_j^\parallel)^\dagger \dot{L}_{j-1}\dots\dot{L}_i x_i^\parallel.$$

**Proof:** We decompose $x_0 = x_0^\parallel + x_0^\perp$. Focusing on $x_0^\perp$ we see that, by definition,

$$w_s^\dagger \dot{L}_{s-1}\dots\dot{L}_0 x_0^\perp = w_s^\dagger \dot{L}_{s-1}\dots\dot{L}_1 x_1.$$

We continue by decomposing $x_1, x_2, \dots$ and eventually this results in:

$$w_s^\dagger \dot{L}_{s-1}\dots\dot{L}_0 x_0 \;=\; w_s^\dagger x_s^\perp + \sum_{i=0}^{s} w_s^\dagger \dot{L}_{s-1}\dots\dot{L}_i x_i^\parallel.$$

We now go through a similar procedure with $w_s^\dagger \dot{L}_{s-1}\dots\dot{L}_i x_i^\parallel$.

$$(w_s^\dagger \dot{L}_{s-1})^\dagger = \dot{L}_{s-1}^\dagger w_s = (\dot{\Pi}\dot{G}_{s-1}\tilde{H})^\dagger w_s = \tilde{H}\dot{G}_{s-1}\dot{\Pi}w_s.$$

Notice that we have used $\tilde{H}^\dagger = \tilde{H}$ (because $H$ is an undirected graph), $\dot{G}_{s-1} = \dot{G}^\dagger_{s-1}$ (because $\dot{G}_i$ is undirected because it implements an involution) and $\tilde{H} = \tilde{H}^\dagger$ (because $H$ is a projection).

We decompose $\dot{G}_{s-1}\dot{\Pi}w_s$ to its parallel and perpendicular parts. The parallel part gives the parallel component $p_{s-1}$ and we stop decomposing it. The perpendicular part gives $w_{s-1}$. I.e.,

$$\begin{aligned}
w_s^\dagger \dot{L}_{s-1}\dots\dot{L}_i x_i^{||} &= (\dot{L}^\dagger_{s-1}w_s)^\dagger \dot{L}_{s-2}\dots\dot{L}_i x_i^{||} \\
&= (\tilde{H}\dot{G}_{s-1}\dot{\Pi}w_s)^\dagger \dot{L}_{s-2}\dots\dot{L}_i x_i^{||} \\
&= (\tilde{H}(\dot{G}_{s-1}\dot{\Pi}w_s)^{||})^\dagger \dot{L}_{s-2}\dots\dot{L}_i x_i^{||} + (\tilde{H}(\dot{G}_{s-1}\dot{\Pi}w_s)^{\perp})^\dagger \dot{L}_{s-2}\dots\dot{L}_i x_i^{||} \\
&= p^\dagger_{s-1}\dot{L}_{s-2}\dots\dot{L}_i x_i^{||} + w^\dagger_{s-1}\dot{L}_{s-2}\dots\dot{L}_i x_i^{||}
\end{aligned}$$

Continuing like this, and noticing that $w_i \perp x_i^{||}$ for $i < s$ because $w_i \in \mathcal{V}^\perp$ and $x_i^{||} \in \mathcal{V}^{||}$, we get:

$$w_s^\dagger \dot{L}_{s-1}\dots\dot{L}_0 x_0 = w_s^\dagger x_s^\perp + w_s^\dagger x_s^{||} + \sum_{s-1\ge j>i\ge 0} p_j^\dagger \dot{L}_{j-1}\dots\dot{L}_i x_i^{||}. \tag{7}$$

■

We now bound each term in the expression,

- $\left|w_s^\dagger x_s\right| \le \|x_s\| \le \lambda_2^s$.
- Next, $\left|\sum_{i=0}^{s-1} p_i^\dagger x_i^{||}\right| \le \sum_{i=0}^{s-1}\|p_i\|\cdot\left\|x_i^{||}\right\| \le \sum_{i=0}^{s-1}\lambda_2^{s-i-1}\lambda_2^i = s\lambda_2^{s-1}$.
- Finally, we are left with the term

$$\sum_{s-1\ge j>i\ge 0} p_j^\dagger \dot{L}_{j-1}\dots\dot{L}_i x_i^{||}. \tag{8}$$

$p_j, x_i^{||} \in \mathcal{V}^{||}$ and therefore $p_j = p_j^{(1)} \otimes \mathbf{1}$, $x_i^{||} = (x_i^{||})^{(1)} \otimes \mathbf{1}$. By Theorem 27, assuming $H$ is $\zeta$-pseudorandom for $G$,

$$\begin{aligned}
p_j^\dagger \dot{L}_{j-1}\dots\dot{L}_i x_i^{||} &\le p_j^{(1)}(\Pi G)^{j-i}(x_i^{||})^{(1)} + \zeta\left\|x_i^{||}\right\|\|p_j\| \\
&\le (\left\|(\Pi G)^{j-i}\right\| + \zeta)\left\|x_i^{||}\right\|\|p_j\| \\
&\le ((\varepsilon_0 + 2\beta + 2\lambda_1)^{\lfloor \frac{j-i}{2}\rfloor} + \zeta)\lambda_2^{s-(j-i)-1} \\
&\le (\lambda_2^{2(\frac{j-i}{2}-1)} + \zeta)\lambda_2^{s-(j-i)-1} \le \lambda_2^{s-3} + \zeta,
\end{aligned}$$

We have used Theorem 15(4) and the assumption that $\varepsilon_0 + 2\beta + 2\lambda_1 \le \lambda_2^2$. Therefore, the term in Equation (8) is bounded by at most $s^2(\lambda_2^{s-3} + \zeta)$.

■

## 7 $H$ is pseudo-random with respect to $\phi$

Now we prove that if $H$ is a Cayley graph over $[D_1]^s$ then it respects the additive structure of $[D_1]^s$ and as a result it is $\zeta = 0$ pseudorandom with respect to $G$. As explained in the introduction, the results in this section were suggested to us by an anonymous referee [17]. We claim:

**Lemma 29.** *Let $G$ and $\phi$ as above. Suppose $D_1 = 2^m$ for some integer $m$, and identify $V_2 = [D_1]^s$ with the additive group $\mathbb{Z}_2^{ms}$. Let $H = Cay(V_2 = \mathbb{Z}_2^{ms} = [D_1]^s, A)$ for some set of generators $A \subseteq \mathbb{Z}_2^{ms} = [D_1]^s$. Then $H$ is $\zeta = 0$ pseudorandom with respect to $G$.*

**Proof:** $H$ is a regular, undirected graph with degree $D_2 = |A|$. Express $H = \frac{1}{D_2}\sum_{j\in[D_2]} \mathcal{H}_j$ where $a_j$ is the $j$'th element in $A$ and $\mathcal{H}_j$ is the transition matrix of the permutation $\gamma_j : \mathbb{Z}_2^{ms} \to \mathbb{Z}_2^{ms}$ defined by $\gamma_j(u) = u + a_j$. Fix any $j_0, \ldots, j_{s-1} \in [D_2]^s$. Then:

**Claim 30.** *For every $0 \le i < s$ and every $u^{(2)} = (u_0^{(2)}, \ldots, u_{s-1}^{(2)}) \in V_2 = [D_1]^s$,*

$$\sigma_{j_0,\ldots,j_i}(u^{(2)}) \;=\; (\tau_0(u_0^{(2)}), \ldots, \tau_{s-1}(u_{s-1}^{(2)}))$$

*for some permutations $\tau_0, \ldots, \tau_{s-1}$ over $[D_1]$.*

**Proof:** By induction over $i$. The case $i = 0$ is clear because

$$\sigma_{j_0}(u^{(2)}) \;=\; \gamma_{j_0}(u^{(2)}) \;=\; u^{(2)} + a_{j_0} = (u_0^{(2)} + (a_{j_0})_0, \ldots, u_{s-1}^{(2)} + (a_{j_0})_{s-1}).$$

Let us assume for $\ell - 1$ and prove for $\ell$. By definition,

$$\sigma_{j_0,\ldots,j_\ell}(u^{(2)}) \;=\; \gamma_{j_\ell}(\psi_{\ell-1}(\sigma_{j_0,\ldots,j_{\ell-1}}(u^{(2)}))).$$

However, by induction, $\sigma_{j_0,\ldots,j_{\ell-1}}$ acts on each coordinate separately. Also, by definition $\psi_\ell$ acts on each coordinate separately. Finally, as in the base case, $\gamma_{j_\ell}$ also acts on each coordinate separately. Hence the induction follows. ■

It follows that $(\pi_0(\sigma_{j_0}(u)), \ldots, \pi_{s-1}(\sigma_{j_0,\ldots,j_{s-1}}(u^{(2)}))) = (\tau_0(u_0^{(2)}), \ldots, \tau_{s-1}(u_{s-1}^{(2)}))$ for some permutations $\tau_0, \ldots, \tau_{s-1}$. Since $u_0^{(2)}, \ldots, u_{s-1}^{(2)}$ are uniform and independent,

$$\left|\pi_0(\sigma_{j_0}(u^{(2)})) \circ \ldots \circ \pi_{s-1}(\sigma_{j_0,\ldots,j_{s-1}}(u^{(2)})) - U_{[D_1]^s}\right|_1 \;=\; 0.$$

■

## 8 Explicitness

### 8.1 Finding primes and irreducible elements.

For many constructions it is necessary to find prime numbers and/or irreducible polynomials over a finite field. For irreducible polynomials the problem is essentially solved:

**Theorem 31.** *[21] There exists an algorithm that given $\ell$ runs in time poly$(\ell)$ and outputs a monic, irreducible polynomial $f \in \mathbb{F}_2[x]$ of degree $\ell$.*

For prime numbers much less is known. The most trivial solution is a brute force search:

**Lemma 32.** *There exists an algorithm that given $n$ runs in time poly$(n)$ and outputs a prime number $n'$ in the interval $[n - n^{0.8}, n]$.*

**Proof:** Let $p_k$ denote the $k$'th prime number and $g_k = p_{k+1} - p_k$ the length of the gap between the $k$'th prime number and the next one. It is known that for every $\alpha > 0$, $g_k < p_k^{3/4+\alpha}$ [23]. Since $p_k = \Theta(\frac{k}{\log k})$ we see that given an integer $n$ there is always a prime $n'$ as requested by the lemma. We can therefore find $n'$ by a brute force search: try all integers in the interval $[n - n^{0.8}, n]$ and for each one of them check primality either by the polynomial time algorithm or by a brute force search. Clearly, the running time is polynomial in $n$. ■

We remark that it is conjectured that $g_k = \text{poly}\log(k)$. If true, we can find $n'$ in a much shorter interval and also can reduce the running time to poly $\log(n)$ by using the polynomial time primality testing algorithm [1].

For the Ramanujan graph construction we will need to find a prime on a given arithmetic progression. If we want to employ the same strategy for finding such a prime, we need to study how large a prime gap on an arithmetic progression can be. Let $\pi_{a,m}(x)$ denote the number of primes $p \le x$ on the arithmetic progression $a + m\mathbb{N}$. Dirichlet's theorem on arithmetic progressions states that if $a$ is co-prime with $m$, then $\pi_{a,m}(x)$ converges to the "correct" bound $\frac{x}{\phi(m)\ln x}$, where $\phi(m)$ is the number of integers co-prime with $m$. This means, for example, that in the limit the density of primes $p$ where $p \equiv 1 (\bmod\ 4)$ is the same as the density of primes $p \equiv 3 (\bmod\ 4)$, hence both these densities tend to $\frac{x}{2\ln x}$. Let $p_n$ be the $n$'th prime number on the arithmetic progression $a + m\mathbb{Z}$. Let $g_n = p_{n+1} - p_n$ be the gap between the $n+1$'th prime and the $n$'th prime. The trivial bound is,

**Claim 33.** *Let $a, m, p_n, g_n$ be as above and $(a, m) = 1$. For every constant $\beta < 1$ and every $n$ large enough, $g_n \le \beta p_n$.*

**Proof:** By Dirichlet's theorem from some $x_0$, $|\pi_{a,m}(x) - \frac{x}{\phi(m)\ln x}| \le \frac{\beta}{4}\frac{x}{\phi(m)\ln x}$ for all $x \ge x_0$. Suppose there is a gap $g_n > \beta p_n$ for some $p_n \ge x_0$. This means

$$(1 - \frac{\beta}{4})(1+\beta)\frac{p_n}{\phi(m)\ln p_n} \le \pi_{a,m}((1+\beta)p_n) = \pi_{a,m}(p_n) \le (1 + \frac{\beta}{4})\frac{p_n}{\phi(m)\ln p_n},$$

a contradiction. ■

With the above claim we have:

**Lemma 34.** *For every $\beta > 0$, there exists an algorithm that given $a, m, n$ such that $(a, m) = 1$ runs in time poly$(n)$ and outputs a prime number $n'$ on the sequence $a + m\mathbb{N}$ and in the interval $[(1-\beta)n, n]$.*

## 8.2 AGHP and Justensen

For the AGHP construction:

**Proof:** (of Lemma 6) We repeat the AGHP construction: Let $q = 2^\ell$ be the first power of 2 such that $q \ge \frac{k}{\varepsilon}$ (so $q < \frac{2k}{\varepsilon}$) and $\mathbb{F}_q$ the field with $q$ elements. The distribution $D$ has $q^2$ elements, the element indexed by $(a,b) \in \mathbb{F}_q^2$ is $(\langle a^0, b\rangle, \langle a^1, b\rangle, \ldots, \langle a^{k-1}, b\rangle) \in \{0,1\}^k$, where the inner product is in $\mathbb{F}_2^\ell$ and is done modulo 2. [3] give the simple proof that $D$ is $\varepsilon$–biased.

For explicitness, given $k$ and $\varepsilon$ we compute $q = 2^\ell$. We then use Lemma 31 to find a degree $\ell$ irreducible polynomial over $\mathbb{F}_2$. We do arithmetic in $\mathbb{F}_q$ modulo this polynomial. We can compute $\langle a^j, b\rangle$ in polylogarithmic time using fast exponentiation. Hence, the construction is *fully explicit.* ■

For the Justensen code:

**Proof:** (of Lemma 7) For $\Upsilon_0$ we take a uniform distribution over the rows of the generating matrix of a Justensen code $[n = ck, k, \alpha]_2$, when $c$ is large enough so that $\frac{k}{n}$ is small enough and the construction works. The set of rows of the generating matrix of the code is a $2(\frac{1}{2} - \alpha)$ biased set. What we still need to check is that $\Upsilon_0$ is fully explicit, or, equivalently, that the code is explicit, i.e., given $i \in [n]$ and $j \in [k]$ we can generate the entry $(i,j)$ of the generating matrix in time $poly(n)$.

Justensen code takes a message word $(\alpha_0, \ldots, \alpha_{k-1}) \in \mathbb{F}^k$ over some field $\mathbb{F}$ of characteristic 2 and size $\Theta(n)$, and interprets it as a polynomial $p_\alpha \in \mathbb{F}[x]$ where $p_\alpha(x) = \sum_{i=0}^{k-1} \alpha_i x^i$. The codeword, when written over $\mathbb{F}$, is $(p_\alpha(q_1), q_1 p_\alpha(q_1), \ldots, p_\alpha(q_\ell), q_\ell p_\alpha(q_\ell))$ where $q_1, \ldots, q_\ell$ is an enumeration of $\mathbb{F}^*$. Thus, the entries in the generating matrix, when taken over $\mathbb{F}$, are $q_i^j$ (as in RS codes) plus rows with entries $q_i q_i^j = q_i^{j+1}$. It follows that all we need for full explicitness is the ability to perform arithmetics in the field $\mathbb{F}$. For that we use Lemma 31. ■

## 8.3 Explicit Expanders

We first describe a family $\{G_{p,q}\}$ of graphs from [14]. We need a (consequence of a) theorem of Jacobi:

**Fact 35.** *Let $p$ be a prime. The are exactly $p+1$ ways to represent $p$ as a sum $a_0^2 + a_1^2 + a_2^2 + a_3^2$, where $a_0$ is odd and $a_1, a_2, a_3$ are even. Let us denote this set of solutions $A_p$. $A_p \subseteq \mathbb{Z}^4$ and $|A_p| = p+1$.*

Assume: $p$ and $q$ are unequal primes, and $p, q \equiv 1 (\mathrm{mod}\ 4)$. The graph $G_{p,q} = (V, E)$ is

$$Cay(PGL(2, \mathbb{F}_q), S_p)$$

where

$$S_p = \left\{ \begin{pmatrix} a_0 + ia_1 & a_2 + ia_3 \\ -a_2 + ia_3 & a_0 - ia_1 \end{pmatrix} : (a_0, a_1, a_2, a_3) \in A_p \right\},$$

and $i$ is a square root of $-1$ in $\mathbb{F}_q$ (see also [12, Section 5]). It turns out this graph is undirected. [14] prove the connected component of the vertex $v_{id}$, corresponding to the identity $e$ of $G$, is Ramanujan. If $\left(\frac{p}{q}\right) = -1$ the connected component is the full graph, and since $|PGL(2,q)| = (q-1)q(q+1)$ we see $G$ is a

$$(N = (q-1)q(q+1), D = p+1, \bar{\lambda} = \frac{2\sqrt{D-1}}{D})$$

graph. If $\left(\frac{p}{q}\right) = 1$ the connected component is half the graph, and the graph is, in fact, a $Cay(PSL(2, \mathbb{F}_q), S_p)$. We see then that $G_{p,q}$ is $(N = \frac{(q-1)q(q+1)}{2}, D = p+1, \bar{\lambda} = \frac{2\sqrt{D-1}}{D})$ graph.

We repeat the lemma we want to prove:

**Lemma 36.** *(Lemma 12 repeated) For every $\beta > 0$, there exists an algorithm that given $n$ and $0 < \bar{\lambda} < 1$ runs in time poly$(n)$ and outputs a Ramanujan graph $G$ with $D \leq \frac{8}{\bar{\lambda}^2}$ and $\bar{\lambda}(G) \leq \bar{\lambda}$. The number of vertices of $G$ is either in the range $[(1-\beta)n, n]$ or in the range $[(1-\beta)2n, 2n]$. The algorithm also outputs a local inversion function $\phi : [D] \to [D]$ computable in polynomial time in its length.*

**Proof:** Find a prime $q \equiv 1(\mod 4)$ in the interval $[((1-\beta)2n)^{1/3}, (2n)^{1/3}]$ and another prime $p \equiv 1(\mod 4)$ in the interval $[(1-\beta)\frac{8}{\bar{\lambda}^2}, \frac{8}{\bar{\lambda}^2}]$. Output the graph $G_{p,q}$ defined above. As the graph is Ramanujan $\bar{\lambda}(G) \leq \frac{2}{\sqrt{D}} \leq \bar{\lambda}$. The running time is poly$(p,q)$ for finding $p, q$ the set $A_p$ and then writing down the whole graph. ■

### 8.3.1 Getting Full explicitness?

Reingold, Vadhan and Wigderson [19] used the zig-zag product to get a combinatorial construction of a fully explicit family of degree $D$ expanders and $\bar{\lambda} \leq D^{-3/4}$. They used a partial derandomization to get $\bar{\lambda} \leq D^{-3/4}$. The $s$-width random walk was originally designed in [6] to give a variant of their construction that is also fully explicit and has $\bar{\lambda} \leq D^{-1/2+o(1)}$. The constructions above are usually sparse in the input length, i.e., they usually construct graphs $G_{n,D}$ for a sparse sequence of lengths $n$ (The [6] construction can handle any degree $D$). However, it seems this is not an inherent limitation, and there should be a construction that can at least handle any smooth integer $n$ (i.e., an integer that factors into a product of small primes).

We remark that can also bypass the whole problem by using Abelian Cayley graphs for $H$. This is inessential if $D$ is large anyway (which is a typical case). This gives a fully explicit construction for slightly sub-constat $\varepsilon$. We believe the work described in this section should lead to a fully explicit construction for the whole range of parameters.

## Acknowledgements

The idea for this paper originated after seeing the $\varepsilon$-bias construction of Eyal Rozenman and Avi Wigderson as described in Andrej Bogdanov's lecture notes [8]. I thank Eyal and Avi for letting me use their unpublished results. I thank Andrej Bogdanov, Arbel Peled and Alon Rosen for introducing me to this construction. I also thank Avi Ben-Aroya, Andrej Bogdanov, Dean Doron, Arbel Peled and Alon Rosen for many interesting discussions about the topic in general and this work in particular, and Dean Doron for detailed comments and suggestions for a first a version of this paper. I owe special thanks for the anonymous referee of the STOC 2008 paper "A combinatorial construction of almost-Ramanujan graphs using the zig-zag product" with Avi Ben-Aroya. His/her suggestions greatly simplify the construction and also improve the parameters the construction achieves.

## References

[1] M. Agrawal, N. Kayal, and N. Saxena. Primes is in $p$. *Annals of mathematics*, pages 781–793, 2004.

[2] N. Alon, J. Bruck, J. Naor, M. Naor, and R. Roth. Construction of asymptotically good, low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory*, 38:509–516, 1992.

[3] N. Alon, O. Goldreich, J. Håstad, and R. Peralta. Simple constructions of almost $k$–wise independent random variables. *Random Structures and Algorithms*, 3(3):289–303, 1992.

[4] N. Alon, A. Lubotzky, and A. Wigderson. Semi-direct product in groups and zig-zag product in graphs: connections and applications. In *Proceedings of the 42nd FOCS*, pages 630–637, 2001.

[5] N. Alon, O. Schwartz, and A. Shapira. An elementary construction of constant degree expanders. In *SODA*, pages 454–458, 2007.

[6] A. Ben-Aroya and A. Ta-Shma. A combinatorial construction of almost-ramanujan graphs using the zig-zag product. *SIAM Journal on Computing*, 40(2):267–290, 2011.

[7] A. Ben-Aroya and A. Ta-Shma. Constructing small-bias sets from algebraic-geometric codes. *Theory of Computing*, 9(5):253–272, 2013.

[8] A. Bogdanov. A different way to improve the bias via expanders. Topics in (and out) the theory of computing, Lecture 12, 2012.

[9] M. Capalbo, O. Reingold, S. Vadhan, and A. Wigderson. Randomness conductors and constant-degree expansion beyond the degree / 2 barrier. In *STOC*, pages 659–668, 2002.

[10] I. Dinur. The pcp theorem by gap amplification. *Journal of the ACM (JACM)*, 54(3):12, 2007.

[11] D. Gillman. A chernoff bound for random walks on expander graphs. *SIAM Journal on Computing*, 27(4):1203–1220, 1998.

[12] S. Hoory, N. Linial, and A. Wigderson. Expander graphs and their applications. *Bulletin of the AMS*, 43(4):439–561, 2006.

[13] J. Justesen. Class of constructive asymptotically good algebraic codes. *IEEE Transactions on Information Theory*, 18(5):652–656, 1972.

[14] A. Lubotzky, R. Philips, and P. Sarnak. Ramanujan graphs. *Combinatorica*, 8:261–277, 1988.

[15] J. Naor and M. Naor. Small–bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993.

[16] A. Nilli. On the second eigenvalue of a graph. *Discrete Mathematics*, 91(2):207–210, 1991.

[17] Anonymous Referee. Reperee report. Private communication from the STOC 2009 committee, 2009.

[18] O. Reingold. Undirected connectivity in log-space. *Journal of the ACM (JACM)*, 55(4):17, 2008.

[19] O. Reingold, S. Vadhan, and A. Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders. *Annals of Mathematics*, 155(1):157–187, 2002.

[20] E. Rozenman and S. Vadhan. Derandomized squaring of graphs. In *RANDOM*, pages 436–447, 2005.

[21] V. Shoup. New algorithms for finding irreducible polynomials over finite fields. *Mathematics of Computation*, 54(189):435–447, 1990.

[22] M. Sudan. Algorithmic introduction to coding theory, lecture 6. http://people.csail.mit.edu/madhu/FT01/scribe/lect6.ps, 2001.

[23] N. Tchudakoff. On the difference between two neighbouring prime numbers. *Rec. Math. [Mat. Sbornik] NS*, 1(6):799–814, 1936.