EXPANDER GRAPHS, WITH EXTENSIONS TO EXPANDER CODES AND DERANDOMIZATION

SHAMIK KHOWALA

1. Abstract

Expander graphs—highly connected but sparse networks—are remarkable objects lying at the intersection of probability, combinatorics, and algebra, with widespread applications across theoretical computer science. We introduce the theory of expander graphs, structured around combinatorial, algebraic, and probabilistic perspectives. We explore the evolution of explicit expander constructions, from breakthrough algebraic structures to the combinatorial zig-zag product. With this foundation, we examine two major algorithmic applications. First, coding theory: how expansion of certain graphs structurally enables coveted linear-time decoding in Sipser-Spielman expander codes. Then, we examine random walks on expander graphs, demonstrating how their properties can be utilized to exponentially reduce the error of randomized algorithms (such as those in **RP** and **BPP** complexity classes). Error-correcting codes and derandomization are vital for security and improved performance in modern computing. Finally, we briefly survey more advanced applications and current research before providing readers with references for further reading.

2. Introduction

Expander graphs have two desired properties that make them useful in analyzing complex networks and designing optimal computational frameworks: *sparsity* and *connectivity*. Sparse graphs have a relatively low number of edges. On the other hand, a graph is 'more' connected if splitting it into many isolated components and disconnecting many pairs of vertices would require more edges to be removed. However, intuitively, if a graph contains very few edges, one expects it to be easily fractured into isolated components by removing a few edges. Conversely, higher connectivity (translating to efficient communication between regions of a network) typically demands a high density of edges. Thus these two properties seem fundamentally opposing. However, as we will see, we can construct graphs that reasonably satisfy both these requirements.

Expander graphs were named as such by Pinsker in 1973 in the context of network concentrators; however, their existence had previously been proven by Kolmogorov and Barzdin in 1967 using the Probabilistic Method: sampling a random d-regular bipartite graph and proving it will have certain expansion properties with high probability [PF73]. However, theoretical computer science heavily restricts the use of randomized existence proofs, because algorithms require explicit, deterministic constructions on which to run. The first explicit construction of expander graphs was achieved shortly after by Margulis, who proved the robust expansion of his algebraic structure using representation theory [Mar73]. The

Date: July 23, 2026.

proof of Margulis' construction was then improved by Gabber and Galil, who used geometric methods. They then used expander graphs to construct linear-size superconcentrators [GG81].

In the 1980s, applications of expander graphs exploded across various domains of discrete mathematics. Tanner pioneered the use of highly connected bipartite graphs in error-correcting codes, a concept that would later play a pivotal role in information theory [Tan81]. Concurrently, expanders were used to construct optimal $O(n \log n)$-size, $O(\log n)$-depth sorting networks [AKS83]. A few years later, the same authors formalized the "hitting property" of random walks on expander graphs, showing how these walks could be used to save random bits in space-bounded computation [AKS87].

By around 1990, the Alon-Boppana bound established the absolute maximum spectral gap any sparse graph could achieve, providing an algebraic limit on expansion [Nil91]. Graphs meeting this optimal bound, known as Ramanujan graphs, were explicitly constructed using complex number theory [LPS88].

The 1990s and 2000s saw more research into expanders combinatorially, and more applications of them in complexity theory. Impagliazzo and Zuckerman utilized expander walks for randomness-efficient error reduction [IZ89]. A few years late, another paper used expander walks for the space-bounded derandomization of pseudorandom generators (PRGs) [INW94]. In coding theory, Sipser and Spielman achieved a historic breakthrough by utilizing bipartite expander graphs to create asymptotically good codes with strictly linear-time decoding algorithms [SS96]. Zémor improved on their work by providing an improved decoding approach to refine the error bound [Zem01].

Perhaps the best development in expander constructions occurred with the introduction of the zig-zag product, which allowed families of expander graphs to be built purely combinatorially in an inductive manner [RVW02]. This same construction was used by Reingold to prove that undirected $s - t$ connectivity belongs to the complexity class $\mathbf{L}$, implying that $\mathbf{L} = \mathbf{SL}$: this demonstrates that symmetric, nondeterministic computations require no more memory than standard deterministic computations [Rei08]. Soon after, Dinur provided a simplified, purely combinatorial proof of the PCP Theorem by using random walks on expanders [Din07].

Expander graphs are still widely applicable today in current research and applications, as discussed in Section 8.

3. Triad of Expansion Definitions

Throughout this paper, unless explicitly stated otherwise, we have the following convention: all graphs $G = (V, E)$ are undirected multi-graphs over a finite set of vertices V of size $|V| = n$. In particular, we permit self-loops and multiple parallel edges. Every graph is assumed to be d-regular, meaning that every single vertex $v \in V$ has a degree d. Finally, for the purpose of random walks and spectral theory, we will consider each undirected edge as a pair of opposing directed edges.

3.1. Combinatorial Expansion: Edge and Vertex Boundaries.

The most intuitive way to define an expander graph is combinatorially. If we select an arbitrary subset of vertices $S \subset V$, we want to ensure that it cannot isolate itself from the remainder of the graph. Formally, we track the number of edges exiting out of S into its complement $\overline{S} = V \backslash S$.

Definition 3.1 (Edge Boundary)**.** For a graph $G = (V, E)$ and a subset of vertices $S \subseteq V$, the *edge boundary* of S, denoted by ∂S, is the set of all edges with one endpoint inside S and one endpoint inside the complement $\overline{S}$:

$$\partial S = E(S, \overline{S}) = \Big\{ \{u, v\} \in E \;\Big|\; u \in S,\ v \in \overline{S} \Big\}. \tag{3.1}$$

If a graph contains a severe bottleneck (which is an area of low connectivity), there will exist a large cluster of vertices S that possesses a very small edge boundary ∂S. Thus to evaluate this uniformly across graphs of varying sizes, we normalize the size of a set's boundary relative to the size of the set itself. Moreover, if a set S grows larger than half the size of the graph, its complement $\overline{S}$ becomes the smaller set, and its boundary size $|\partial S|$ naturally decreases relative to $|S|$. Thus we only consider subsets with at most half of the total vertices ($|S| \leq \frac{n}{2}$). Now we arrive at the definition of expansion in terms of edges:

Definition 3.2 (Edge Expansion Ratio)**.** The *edge expansion ratio* (also referred to as the Cheeger constant or the isoperimetric number) of a graph G, denoted by $h(G)$, is defined as the minimum ratio of a set's edge boundary to its size, taken over all non-empty subsets comprising at most half the vertices in the graph:

$$h(G) = \min_{\substack{S \subseteq V \\ 0 < |S| \leq \frac{n}{2}}} \frac{|\partial S|}{|S|}. \tag{3.2}$$

For a single finite graph, $h(G)$ is simply a fixed value. The true power of expander graphs emerges when we analyze a *collection* of graphs whose sizes grow arbitrarily large:

Definition 3.3 (Family of Expander Graphs)**.** A sequence of d-regular graphs $\{G_i\}_{i \in \mathbb{N}}$ of strictly increasing vertex sizes $n_i \to \infty$ is defined as a *family of expander graphs* if there exists a fixed constant $\epsilon > 0$, independent of i, such that the edge expansion ratio of every graph in the sequence satisfies:

$$h(G_i) \geq \epsilon \quad \text{for all } i \in \mathbb{N}. \tag{3.3}$$

Alternatively, expansion can be framed around the *vertex boundary* rather than edges. Let $N(S) = \{v \in V \mid \exists u \in S \text{ such that } \{u, v\} \in E\}$ be the neighborhood of a vertex set S. The vertex boundary expansion, similarly to $h(G)$, can be found as

$$\frac{|N(S) \setminus S|}{|S|}.$$

3.2. **Algebraic Expansion: Spectral Graph Theory.** While combinatorial expansion is more simply defined, actually calculating $h(G)$ is quite difficult; finding the exact minimum cut over all possible subsets is an NP-hard problem. Spectral graph theory bypasses this difficulty by instead analyzing the graph through linear algebra.

Let $G = (V, E)$ be a d-regular graph with n vertices, and let $A \in \mathbb{R}^{n \times n}$ denote its standard adjacency matrix, defined entry-wise by:

$$A_{ij} = \text{the number of edges connecting vertex } i \text{ and vertex } j. \tag{3.4}$$

Because G is undirected, A is a real, symmetric matrix. Thus, by the Spectral Theorem, A has n real eigenvalues

$$\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_n. \tag{3.5}$$

Furthermore, there exists an orthonormal system of eigenvectors $\{v_1, v_2, \ldots, v_n\}$ corresponding to these eigenvalues. We first establish the properties of λ_1:

Theorem 3.4. *Let G be a d-regular graph with adjacency matrix A. The largest eigenvalue is exactly $\lambda_1 = d$, and its corresponding normalized eigenvector is the all-ones vector scaled to unit length:*

$$v_1 = \frac{1}{\sqrt{n}}\mathbf{1} = \left(\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}}\right)^T. \tag{3.6}$$

Proof. First, we show that $\mathbf{1} = (1, 1, \ldots, 1)^T$ is an eigenvector with eigenvalue d. Consider the i-th entry of the product $A\mathbf{1}$:

$$(A\mathbf{1})_i = \sum_{j=1}^{n} A_{ij} \cdot 1 = \sum_{j=1}^{n} A_{ij}. \tag{3.7}$$

Since the graph is d-regular, the sum of the entries in row i is exactly equal to the degree of vertex i, which is d. Thus, $(A\mathbf{1})_i = d \cdot 1$ for all i, meaning $A\mathbf{1} = d\mathbf{1}$ as desired. Normalizing this vector to have an ℓ_2 norm of 1 yields $v_1 = \frac{1}{\sqrt{n}}\mathbf{1}$.

To prove that no eigenvalue can have an absolute value strictly greater than d, let x be any arbitrary eigenvector of A with eigenvalue μ, so $Ax = \mu x$. Let i be the index of the coordinate of x such that $|x_i| \geq |x_j|$ for all j. Looking at the i-th coordinate of the eigenvalue equation:

$$\mu x_i = (Ax)_i = \sum_{j=1}^{n} A_{ij}x_j \iff |\mu| \cdot |x_i| = \left|\sum_{j=1}^{n} A_{ij}x_j\right| \leq \sum_{j=1}^{n} A_{ij}|x_j|. \tag{3.8}$$

Since $|x_j| \leq |x_i|$ for all j, we can upper-bound the right side by substituting $|x_i|$ for $|x_j|$:

$$|\mu| \cdot |x_i| \leq \sum_{j=1}^{n} A_{ij}|x_i| = |x_i| \sum_{j=1}^{n} A_{ij} = d \cdot |x_i|. \tag{3.9}$$

Dividing by $|x_i| > 0$ yields $|\mu| \leq d$, as desired. ∎

The main parameter considered for algebraic expansion is the distance between λ_1 and the rest of the spectrum. We define the *spectral gap* as:

$$\gamma = d - \lambda_2. \tag{3.10}$$

A large spectral gap means λ_2 is tiny compared to d. Typically a large spectral gap implies high expansion due to Cheeger's Inequality:

$$\frac{d - \lambda_2}{2} \leq h(G) \leq \sqrt{2d(d - \lambda_2)}. \tag{3.11}$$

Cheeger's Inequality also ensures that a sequence of graphs forms an expander family if and only if it maintains a strictly positive algebraic spectral gap (the LHS) as $n \to \infty$.

To further demonstrate why a large spectral gap implies better expansion, we examine the Expander Mixing Lemma. This theorem shows that a large spectral gap forces the distribution of edges between any two subsets of vertices to closely match what one would expect to see in a completely random graph (under the Erdős-Rényi model).

Let λ denote the largest absolute value of an eigenvalue other than $\lambda_1 = d$:

$$\lambda = \max_{i \geq 2} |\lambda_i| = \max\left(|\lambda_2|, |\lambda_n|\right). \tag{3.12}$$

For any two subsets of vertices $S, T \subseteq V$, we let $E(S,T)$ denote the set of ordered pairs $(u,v) \in S \times T$ such that there is an edge connecting u and v. If S and T overlap, edges contained within $S \cap T$ are counted twice, reflecting the fact that the walk can move across that edge in both directions.

Theorem 3.5 (Expander Mixing Lemma)**.** *Let $G = (V,E)$ be a d-regular graph on n vertices with parameter λ as defined above. Then for any two subsets $S, T \subseteq V$:*

$$\left| |E(S,T)| - \frac{d|S||T|}{n} \right| \leq \lambda \sqrt{|S||T| \left(1 - \frac{|S|}{n}\right)\left(1 - \frac{|T|}{n}\right)} \leq \lambda \sqrt{|S||T|}. \tag{3.13}$$

Proof. Let $\mathbf{1}_S, \mathbf{1}_T \in \mathbb{R}^n$ be the characteristic indicator vectors of the sets S and T respectively, where the i-th entry is 1 if vertex i belongs to the set, and 0 otherwise. The total number of edges between S and T can be expressed using the adjacency matrix A:

$$|E(S,T)| = \mathbf{1}_S^T A \mathbf{1}_T. \tag{3.14}$$

Let $\{v_1, v_2, \ldots, v_n\}$ be the orthonormal basis of eigenvectors of A, with $v_1 = \frac{1}{\sqrt{n}}\mathbf{1}$. We can uniquely decompose $\mathbf{1}_S$ and $\mathbf{1}_T$ into linear combinations of these eigenvectors:

$$\mathbf{1}_S = \sum_{i=1}^{n} \alpha_i v_i, \qquad \mathbf{1}_T = \sum_{j=1}^{n} \beta_j v_j. \tag{3.15}$$

The coefficients α_i and β_j are found by taking the inner product with the respective basis vectors. For the primary eigenvector v_1, these coefficients are:

$$\alpha_1 = \langle \mathbf{1}_S, v_1 \rangle = \sum_{u \in S} 1 \cdot \frac{1}{\sqrt{n}} = \frac{|S|}{\sqrt{n}}, \tag{3.16}$$

$$\beta_1 = \langle \mathbf{1}_T, v_1 \rangle = \sum_{v \in T} 1 \cdot \frac{1}{\sqrt{n}} = \frac{|T|}{\sqrt{n}}. \tag{3.17}$$

Now, substitute these spectral expansions directly back into the expression for $|E(S,T)|$:

$$|E(S,T)| = \left(\sum_{i=1}^{n} \alpha_i v_i\right)^T A \left(\sum_{j=1}^{n} \beta_j v_j\right) = \left(\sum_{i=1}^{n} \alpha_i v_i\right)^T \left(\sum_{j=1}^{n} \beta_j \lambda_j v_j\right). \tag{3.18}$$

By exploiting the orthonormality of the eigenvectors ($v_i^T v_j = 1$ if $i = j$ and 0 otherwise), all cross-terms vanish completely, simplifying the equation to

$$|E(S,T)| = \sum_{i=1}^{n} \lambda_i \alpha_i \beta_i. \tag{3.19}$$

We separate this summation into the first eigenvalue component and remaining components:

$$|E(S,T)| = \lambda_1 \alpha_1 \beta_1 + \sum_{i=2}^{n} \lambda_i \alpha_i \beta_i = d\left(\frac{|S|}{\sqrt{n}}\right)\left(\frac{|T|}{\sqrt{n}}\right) + \sum_{i=2}^{n} \lambda_i \alpha_i \beta_i. \tag{3.20}$$

$$|E(S,T)| = \frac{d|S||T|}{n} + \sum_{i=2}^{n} \lambda_i \alpha_i \beta_i. \tag{3.21}$$

Displacing terms and bounding each $|\lambda_i|$ by λ, we obtain

$$\left| |E(S,T)| - \frac{d|S||T|}{n} \right| = \left| \sum_{i=2}^{n} \lambda_i \alpha_i \beta_i \right| \leq \sum_{i=2}^{n} |\lambda_i||\alpha_i||\beta_i| \leq \lambda \sum_{i=2}^{n} |\alpha_i||\beta_i|. \tag{3.22}$$

We now apply the Cauchy-Schwarz inequality to the vector of coefficients $(\alpha_2, \ldots, \alpha_n)$ and $(\beta_2, \ldots, \beta_n)$:

$$\sum_{i=2}^{n} |\alpha_i||\beta_i| \leq \sqrt{\sum_{i=2}^{n} \alpha_i^2} \cdot \sqrt{\sum_{i=2}^{n} \beta_j^2}. \tag{3.23}$$

By Parseval's identity, the total squared norm of the coefficients is equal to the squared ℓ_2 norm of the original characteristic vector:

$$\sum_{i=1}^{n} \alpha_i^2 = \|\mathbf{1}_S\|_2^2 = \sum_{u \in S} (1)^2 = |S|. \tag{3.24}$$

Therefore, the summation starting from $i = 2$ can be written by stripping out the first coefficient α_1:

$$\sum_{i=2}^{n} \alpha_i^2 = |S| - \alpha_1^2 = |S| - \frac{|S|^2}{n} = |S| \left(1 - \frac{|S|}{n}\right). \tag{3.25}$$

Applying the exact same identity to the coefficients of T:

$$\sum_{i=2}^{n} \beta_j^2 = |T| - \beta_1^2 = |T| - \frac{|T|^2}{n} = |T| \left(1 - \frac{|T|}{n}\right). \tag{3.26}$$

Substituting these terms back into our Cauchy-Schwarz bound yields:

$$\left| |E(S,T)| - \frac{d|S||T|}{n} \right| \leq \lambda \sqrt{|S| \left(1 - \frac{|S|}{n}\right) \cdot |T| \left(1 - \frac{|T|}{n}\right)}. \tag{3.27}$$

Since $\left(1 - \frac{|S|}{n}\right) \leq 1$ and $\left(1 - \frac{|T|}{n}\right) \leq 1$, we arrive at the final simplified upper bound:

$$\left| |E(S,T)| - \frac{d|S||T|}{n} \right| \leq \lambda \sqrt{|S||T|}. \tag{3.28}$$

■

The term $\frac{d|S||T|}{n}$ represents the exact expected number of edges that would form between S and T if edges were distributed completely at random with uniform density. Thus, the Expander Mixing Lemma reveals that the parameter λ directly governs the maximum possible deviation from randomization. When λ is exceptionally small (indicating a massive spectral gap and high expansion), the graph behaves essentially like a random graph, with a very uniform distribution of edges. This lemma bridges spectral expansion and probabilistic expansion, which we examine next.

3.3. **Probabilistic Expansion: Random Walks.** In our third perspective of expander graphs, we consider random walks. We observe the behavior of a random walk on the graph, analogous to an agent traversing the network. Note that random walks are heavily utilized across computer science, such as in several AI algorithms like reinforcement learning and diffusion models.

We first define the *normalized adjacency matrix* $\hat{A}$ as

$$\hat{A} = \frac{1}{d} A. \tag{3.29}$$

In particular, the matrix $\hat{A}$ possesses several linear algebraic properties:

- It is real and symmetric.
- It is doubly stochastic, meaning that every column and every row sums up exactly to 1.
- Its eigenvalues are $1 = \hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \cdots \geq \hat{\lambda}_n \geq -1$. The eigenvectors remain completely identical to those of A.

A random walk on G is a discrete-time Markov Chain where the state space is the vertex set V. If the current state of the walk is vertex u, the next state is selected uniformly at random from among the d neighbors of u. If we represent the probability distribution of our current position as a column vector $p \in \mathbb{R}^n$ (where $p_i \geq 0$ and $\sum p_i = 1$), transitioning to a random neighbor is mathematically equivalent to multiplying the vector by $\hat{A}$. That is, the distribution at the next step is $p' = \hat{A}p$. Thus, if we initialize the walk from an arbitrary starting distribution p, the probability distribution after t consecutive steps is

$$p^{(t)} = \hat{A}^t p. \tag{3.30}$$

The stationary distribution of this random walk is the uniform distribution: $u = \left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right)^T$. Because $\hat{A}$ is doubly stochastic, the uniform distribution is invariant, meaning $\hat{A}u = u$.

To show that an expander graph mixes rapidly, we must prove that $\hat{A}^t p$ converges exponentially fast to u, regardless of the choice of initial distribution p. We measure the distance between probability distributions using two distinct norms: the ℓ_2 norm (Euclidean distance) and the ℓ_1 norm. We also define α:

$$\alpha = \max_{i \geq 2} |\hat{\lambda}_i| = \frac{\lambda}{d}. \tag{3.31}$$

We now prove that the distance to the uniform vector shrinks by a factor of exactly α at every single step of the walk.

Lemma 3.6. *Let G be a d-regular graph with normalized adjacency matrix $\hat{A}$ and parameter α as defined above. Then for any probability distribution vector $p \in \mathbb{R}^n$:*

$$\|\hat{A}p - u\|_2 \leq \alpha \|p - u\|_2. \tag{3.32}$$

Proof. We express the left-hand side of Equation 3.32 as

$$\hat{A}p - u = \hat{A}p - \hat{A}u = \hat{A}(p - u). \tag{3.33}$$

Let $z = p - u$ representing the deviation from uniformity. We examine the inner product of z with $v_1 = \frac{1}{\sqrt{n}}\mathbf{1}$:

$$\langle z, \mathbf{1} \rangle = \sum_{i=1}^{n} \left(p_i - \frac{1}{n}\right) = \sum_{i=1}^{n} p_i - \sum_{i=1}^{n} \frac{1}{n} = 1 - 1 = 0. \tag{3.34}$$

Thus z is orthogonal to v_1. Therefore, if we decompose z into the orthonormal basis of eigenvectors $\{v_1, v_2, \ldots, v_n\}$, the coefficient corresponding to v_1 must be exactly zero, giving:

$$z = \sum_{i=2}^{n} \gamma_i v_i. \tag{3.35}$$

Applying the linear transformation $\hat{A}$ to this, we obtain:

$$\hat{A}z = \hat{A}\left(\sum_{i=2}^{n} \gamma_i v_i\right) = \sum_{i=2}^{n} \gamma_i \hat{A}_i v_i = \sum_{i=2}^{n} \gamma_i \hat{\lambda}_i v_i. \tag{3.36}$$

Now evaluate the squared ℓ_2 norm of $\hat{A}z$. Since the eigenvectors are orthonormal, all cross-product terms drop out to give:

$$\|\hat{A}z\|_2^2 = \sum_{i=2}^{n}(\gamma_i\hat{\lambda}_i)^2 = \sum_{i=2}^{n}\gamma_i^2\hat{\lambda}_i^2. \tag{3.37}$$

We upper-bound each eigenvalue $\hat{\lambda}_i^2$ by α^2:

$$\|\hat{A}z\|_2^2 \leq \sum_{i=2}^{n}\gamma_i^2\alpha^2 = \alpha^2\sum_{i=2}^{n}\gamma_i^2. \tag{3.38}$$

Evaluating the squared ℓ_2 norm of z gives $\sum_{i=2}^{n}\gamma_i^2 = \|z\|_2^2$, so we obtain:

$$\|\hat{A}z\|_2^2 \leq \alpha^2\|z\|_2^2 \iff \|\hat{A}(p-u)\|_2 \leq \alpha\|p-u\|_2. \tag{3.39}$$

■

We can induct on this lemma to obtain the following fundamental theorem.

Theorem 3.7 (Spectral Convergence Theorem)**.** *Let G be a d-regular graph with normalized adjacency matrix $\hat{A}$ and parameter α. Then for any initial probability vector p and any integer step count $t \geq 1$:*

$$\|\hat{A}^t p - u\|_2 \leq \alpha^t\|p-u\|_2 \leq \alpha^t, \tag{3.40}$$

$$\|\hat{A}^t p - u\|_1 \leq \sqrt{n}\cdot\alpha^t. \tag{3.41}$$

Proof. The ℓ_2 upper bound follows directly by applying Lemma 3.6 inductively t times:

$$\|\hat{A}^t p - u\|_2 = \|\hat{A}(\hat{A}^{t-1}p - u)\|_2 \leq \alpha\|\hat{A}^{t-1}p - u\|_2 \leq \cdots \leq \alpha^t\|p-u\|_2 \leq \alpha^t. \tag{3.42}$$

To derive the ℓ_1 convergence bound, we apply the Cauchy-Schwarz inequality: for any arbitrary vector $x \in \mathbb{R}^n$, $\|x\|_1 \leq \sqrt{n}\|x\|_2$. Setting $x = \hat{A}^t p - u$:

$$\|\hat{A}^t p - u\|_1 \leq \sqrt{n}\|\hat{A}^t p - u\|_2 \leq \sqrt{n}\cdot\alpha^t. \tag{3.43}$$

■

The Spectral Convergence Theorem shows the idea of probabilistic expansion. If a graph has a large spectral gap, then α is strictly less than 1. As the length of our random walk t increase, the term α^t decays exponentially to zero. This ensures that the random walk will completely erase any bias present in its initial distribution and distribute its probability mass evenly across the network in $O(\log n)$ steps. This rapid convergence drives expander walk sampling and derandomization techniques presented in Section 7.

4. Bounds and Computability

As we have discussed, determining the exact expansion of an arbitrary graph is computationally intractable; calculating $h(G)$ requires evaluating the boundary of all possible subsets $S \subset V$, which is NP-hard (quickly verifiable but not efficient to find). Fortunately, spectral graph theory helps to bridge these computational difficulties. Computing the eigenvalues of a matrix belongs to the complexity class **P** (computed in polynomial time) via standard linear algebraic methods. In this section, we explore how the algebraic spectrum bounds the combinatorial geometry of these graphs.

4.1. **Cheeger's Inequality.** The relationship between the combinatorial edge expansion $h(G)$ and the algebraic spectral gap $d - \lambda_2$ is governed by the discrete Cheeger Inequality.

Theorem 4.1 (Discrete Cheeger Inequality)**.** *Let G be a d-regular graph with spectrum $d = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Then the edge expansion ratio $h(G)$ is bounded by the spectral gap:*

$$\frac{d - \lambda_2}{2} \leq h(G) \leq \sqrt{2d(d - \lambda_2)}$$

Both the upper and lower bounds of this inequality are asymptotically tight. The lower bound is tight for the d-dimensional hypercube Q_d, which achieves an expansion $h(G) = 1$ and a spectral gap of $d - \lambda_2 = 2$. The upper bound is tight for the n-vertex cycle graph C_n, which is a 2-regular graph demonstrating an expansion $h(C_n) = \Theta(1/n)$ alongside a spectral gap of $d - \lambda_2 = \Theta(1/n^2)$.

For the proof of the Discrete Cheeger Inequality, refer to Appendix A.

4.2. **The Alon-Boppana Bound.** Cheeger's Inequality reveals that to maximize expansion, we must minimize its second eigenvalue λ_2 (or more generally, the absolute second eigenvalue $\lambda = \max(|\lambda_2|, |\lambda_n|)$). However, there is a mathematical limit to how thoroughly connected a sparse graph can become. As the number of vertices n grows toward infinity, the spectral gap is fundamentally restricted by the degree d via the Alon-Boppana Bound.

Theorem 4.2 (Alon-Boppana Bound)**.** *For every (n, d)-regular graph, the second largest absolute eigenvalue λ satisfies:*

$$\lambda \geq 2\sqrt{d - 1} - o_n(1)$$

The $o_n(1)$ term denotes a quantity that asymptotically approaches zero for any fixed d as $n \to \infty$. This theorem reveals that no sequence of fixed-degree graphs can maintain an absolute eigenvalue strictly smaller than $2\sqrt{d-1}$. Graphs that asymptotically meet this lower limit are termed Ramanujan graphs. They represent the optimal, theoretically perfect expanders. While the full proof of the Alon-Boppana bound is slightly beyond the scope of this paper, we can elegantly prove a slightly weaker limit in the following lemma.

Lemma 4.3 (Trace Bound on Eigenvalues)**.** *For every d-regular graph G on n vertices, $\lambda \geq \sqrt{d}(1 - o_n(1))$.*

Proof. Let A be the adjacency matrix of G. The value of $\text{Trace}(A^2)$ precisely counts the number of length-2 walks that start and end on the same vertex. From any given vertex v, one can always traverse an incident edge and immediately return along the exact same edge. Because v has exactly d incident edges, there are at least d such trivial closed walks. Therefore, the diagonal entries of A^2 are all at least d, meaning:

$$\text{Trace}(A^2) \geq nd.$$

Moreover, it is known that the trace of a squared matrix equals the sum of its squared eigenvalues (in particular, the squares of the eigenvalues of any matrix M are the eigenvalues of the squared matrix M^2), giving:

$$\text{Trace}(A^2) = \sum_{i=1}^{n} \lambda_i^2.$$

We know $\lambda_1^2 = d^2$. The remaining $n-1$ eigenvalues are all bounded by λ^2, allowing us to establish the upper bound:

$$\text{Trace}(A^2) \leq d^2 + (n-1)\lambda^2.$$

Combining our combinatorial lower bound with our algebraic upper bound yields:

$$nd \leq d^2 + (n-1)\lambda^2 \implies \lambda^2 \geq d \cdot \frac{n-d}{n-1}.$$

Taking the square root of both sides gives $\lambda \geq \sqrt{d}$ as n grows large, completing the proof. ■

5. Explicit Constructions

By the Probabilistic Method, we know that if we sample a random d-regular graph, it will be an excellent expander with overwhelmingly high probability. For instance, fixing $d \geq 3$, a uniformly random d-regular graph on n vertices is an $(\Omega(n), d - 1.01)$-vertex expander as $n \to \infty$. However, in practical applications of expander graphs, we require explicit families of expander graphs where the neighborhood of any given vertex can be computed in $\log n$-polynomial time.

5.1. **Algebraic Constructions.**

The earliest successful explicit constructions relied on group theory and number theory.

5.1.1. *The Margulis Construction (1973).*

Margulis provided the very first explicit family of expander graphs. For any integer m, he defined a graph G_m on the vertex set $V_m = \mathbb{Z}_m \times \mathbb{Z}_m$. Each vertex (x, y) is connected to eight neighbors modulo m:

$$(x \pm y, y), \quad (x, y \pm x), \quad (x \pm y + 1, y), \quad (x, y \pm x + 1)$$

This yields an 8-regular graph. While conceptually simple to construct, proving that its expansion ratio $h(G_m)$ is bounded away from zero by an absolute constant requires non-trivial representation theory.

5.1.2. *Prime Inverse Construction [Vad12].*

Another highly elegant, 3-regular algebraic construction derives from the finite field $\mathbb{Z}_p$ for any prime p. The vertex set is $V_p = \mathbb{Z}_p$. A vertex x is connected to:

$$x + 1 \pmod{p}, \quad x - 1 \pmod{p}, \quad x^{-1} \pmod{p}$$

(where the inverse of 0 is defined as 0). See Figure 1 for an example.

5.2. **Graph Products.**

While algebraic constructions are highly optimal, their proofs of expansion are difficult. A major breakthrough in modern theoretical computer science was the discovery of fully combinatorial constructions of expander graphs using iterative graph products. This approach allows us to start with a tiny, trivially verified expander graph and grow it to an arbitrary size while mathematically controlling its expansion.

We denote graphs by the tuple (n, d, α), representing a d-regular graph on n vertices with a normalized second eigenvalue $\alpha = \frac{\lambda}{d} < 1$.

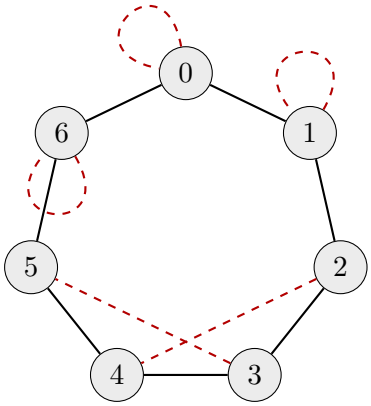

Figure 1. The 3-regular graph on $\mathbb{Z}_7$ with x connected to $x+1$, $x-1$, and $x^{-1} \pmod p$. Solid black edges forming the outer 7-cycle come from $x \mapsto x \pm 1$. Dashed red edges come from $x \mapsto x^{-1} \pmod 7$.

5.2.1. *Graph Squaring.* The simplest operation to increase connectivity is graph squaring. If $G = (V, E)$ is a d-regular graph, its square $G^2 = (V, E')$ is a graph on the same vertex set where an edge $(u, w) \in E'$ exists for every walk of exactly length 2 between u and w in G. See Figure 2. Now G^2 is a d^2-regular graph and its normalized second eigenvalue improves from α to α^2. While squaring exponentially reduces the second eigenvalue, the cost is the exponential loss of sparsity. Squaring alone cannot produce a constant-degree expander family!

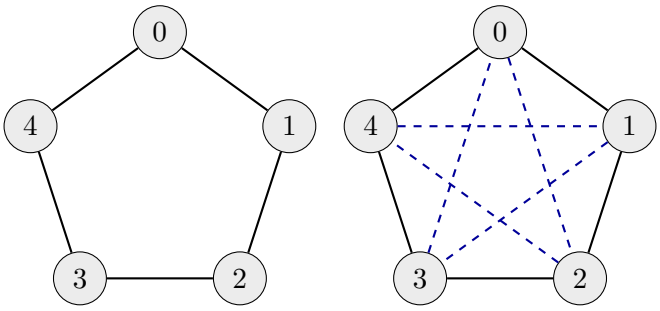

Figure 2. Left: the 2-regular cycle $G = C_5$. Right: G^2, whose edges are the pairs joined by a walk of length 2 in G. Solid edges are inherited from G; dashed blue edges are the new distance-2 connections $i \leftrightarrow i+2 \pmod 5$. Since every non-adjacent pair in C_5 is at distance exactly 2, squaring produces the complete graph K_5 (degree $2^2 = 4$). Each vertex also gains two self-loops from back-and-forth walks $i \to i \pm 1 \to i$, which we omit.

5.2.2. *The Replacement Product.* The replacement product $G \circledR H$ takes a large graph G of n vertices and degree m, and a smaller, highly expanding graph H of m vertices and degree d. The product replaces every single vertex $v \in G$ with a "cloud" of m vertices, each corresponding to an edge incident to v (a perfect matching). Inside each cloud, the m vertices are connected exactly according to the structure of H. The resulting graph $G \circledR H$

has nm vertices and a degree of $d+1$. However, it lacks symmetry between the inter-cloud and intra-cloud edges, which complicates spectral analysis.

5.2.3. *The Zig-Zag Product.* Introduced by Reingold, Vadhan, and Wigderson, the zig-zag product, denoted $G \textcircled{z} H$, combines the replacement product with a length-3 walk to maintain symmetry and bound the spectral gap. Let G be an (n, m, α)-graph and H be an (m, d, β)-graph. The vertex set of $G \textcircled{z} H$ is the Cartesian product $V(G) \times V(H)$, structurally identical to the n clouds of m vertices from the replacement product. An edge exists between (v, i) and (u, j) in $G \textcircled{z} H$ if and only if there is a three-step path consisting of:

(1) A "zig" step in H: a move within cloud v from local node i to local node k using an edge in H,
(2) A step in G: a move across a cross-cloud edge of G from node k in cloud v to node l in cloud u,
(3) A "zag" step in H: a move within cloud u from local node l to local node j using an edge in H,

as shown in Figure 3 below. A partial example is given in Figure 4 below.

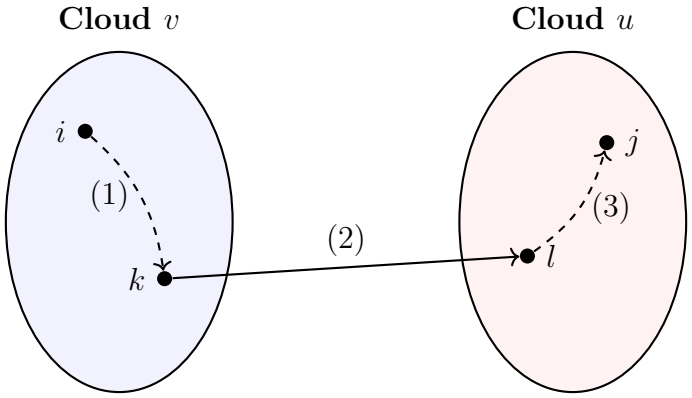

Figure 3. A single edge transition in the zig-zag product $G \textcircled{z} H$.

The resulting graph $G \textcircled{z} H$ is a uniform $(nm, d^2, \varphi(\alpha, \beta))$-graph. Reingold, Vadhan, and Wigderson proved the fundamental bound on the new spectral gap:

$$\varphi(\alpha, \beta) \leq \alpha + \beta + \beta^2.$$

Finally, we alternate the zig-zag product with graph squaring. Squaring a graph drives its spectral gap down (improving expansion) but increases the degree. Zig-zagging it with a small, fixed expander H pulls the degree back down to a constant while only slightly changing the spectral gap.

Let H be a fixed, pre-computed $(d^4, d, 1/8)$-graph. Because its size d^4 is a constant, H can be found in $O(1)$ time via brute-force search. We define our infinite family of expanders $\{G_n\}$ recursively:

$$G_1 = H^2 \tag{5.1}$$

$$G_{n+1} = G_n^2 \textcircled{z} H \quad \text{for } n \geq 1 \tag{5.2}$$

We now prove the following.

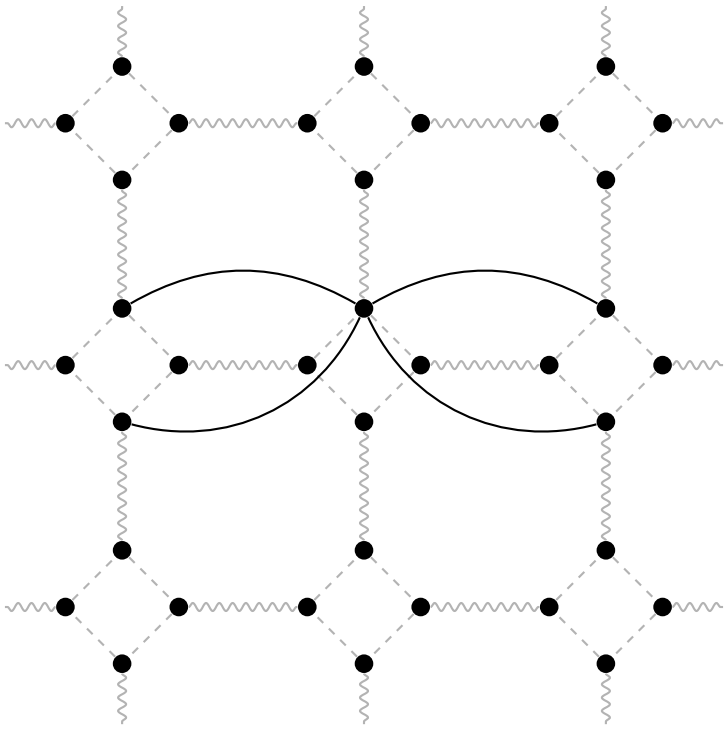

Figure 4. The zig-zag product of the grid $\mathbb{Z}^2$ with the 4-cycle. Some of its edges are shown with solid lines, made from zig-zag walks: dashed to wiggly to dashed. The dashed and wiggly edges are the original edges of the replacement product of the two graphs.

Theorem 5.1. *For all $n \geq 1$, the graph G_n is a $(d^{4n}, d^2, 9/16)$-expander graph.*

Proof. We proceed by induction. For the base case $n = 1$, $G_1 = H^2$. The number of vertices is d^4. The degree is d^2. The spectral bound is $(1/8)^2 = 1/64 \leq 9/16$. Thus, G_1 is a $(d^4, d^2, 9/16)$-graph. Assume inductively that G_n is a $(d^{4n}, d^2, 9/16)$-graph. We compute the parameters for $G_{n+1} = G_n^2 ⓩ H$:

(1) **Squaring G_n:** The graph G_n^2 has d^{4n} vertices, degree d^4, and a spectral bound of $(9/16)^2 = 81/256$.
(2) **Zig-Zag Set-Up:** Because G_n^2 has degree d^4, it perfectly matches the number of vertices in H, making the product valid.
(3) **Applying Zig-Zag Product:** The final vertex count is $(d^{4n})(d^4) = d^{4(n+1)}$. The final degree relies only on H and becomes d^2. By the zig-zag spectral gap bound, the new bound is $\varphi(81/256, 1/8) \leq 81/256 + 1/8 + 1/64 \leq 9/16$.

Therefore, G_{n+1} is a $(d^{4(n+1)}, d^2, 9/16)$-graph, establishing a strongly explicit, constant-degree expander family. ■

6. Applications I: Expander Codes

The theory of expander graphs is a foundational pillar of modern algorithms. One of the many applications of expansion is in the realm of error-correcting codes.

When transmitting information across a noisy channel, bits can be corrupted. The goal of coding theory is to introduce redundancy into a message so that even if a fraction of the

transmitted bits are altered, the original message can still be perfectly reconstructed. To do this, we must design codes that have these three properties:

- **High Rate:** The ratio of actual message bits to transmitted bits should be as close to 1 as possible. In other words, we want to have minimal redundancy.
- **Large Distance:** The code must be able to tolerate a large number of errors.
- **Efficiency:** The algorithms to encode the message and decode the corrupted transmission must run fast: ideally linear time.

Historically, constructing codes with all three properties was quite difficult. However, by using bipartite expander graphs, theoretical computer scientists were able to structurally guarantee these properties, culminating in codes that can be decoded in linear $O(n)$ time. Encoding can theoretically be done in linear $O(n)$ time as well, although practical applications settle for near-linear time.

6.1. **Preliminaries of Linear Codes.** We begin by establishing the fundamental definitions of coding theory over the binary alphabet $\{0, 1\}$.

Definition 6.1 (Binary Code)**.** A binary code C of *block-length* n is simply a subset of all possible n-bit strings, meaning $C \subset \{0,1\}^n$. The elements of C are called *codewords*.

Let's say Alice wants to send a k-bit message to Bob. Then she will use an encoding function $E : \{0,1\}^k \to C$ to map her k-bit message to an n-bit codeword ($n > k$). Bob receives an n-bit string y that may have been corrupted. He must determine which codeword $x \in C$ Alice originally sent.

To measure how distinct two codewords are, we use the Hamming distance.

Definition 6.2 (Hamming Distance)**.** The Hamming distance $d_H(x, y)$ between two strings $x, y \in \{0,1\}^n$ is the number of coordinates in which they differ:

$$d_H(x, y) = |\{i : x_i \neq y_i\}|.$$

The *Hamming weight* of a single string x is the number of 1's it contains.

For Bob to successfully decode the message, valid codewords must be "far apart" from one another. If codewords are too close, a small amount of noise could easily flip one valid codeword into another. This notion motivates the following definitions for codes:

Definition 6.3 (Distance and Rate)**.** The distance of a code C, denoted $\text{dist}(C)$, is the minimum Hamming distance between any pair of distinct codewords:

$$\text{dist}(C) = \min_{\substack{x,y \in C \\ x \neq y}} d_H(x, y).$$

The rate of a code evaluates its efficiency by measuring the proportion of information bits to transmitted bits:

$$\text{rate}(C) = \frac{\log_2 |C|}{n}.$$

If a code has a distance of d, it can safely correct up to $\lfloor (d-1)/2 \rfloor$ errors. If fewer than this number of bits are flipped, the corrupted string y will strictly be closer (in Hamming distance) to the original sent codeword x than to any other valid codeword.

Definition 6.4 (Asymptotically Good Codes)**.** A family of codes $C_n \subset \{0,1\}^n$ is asymptotically good if there exist fixed constants $d > 0$ and $r > 0$ such that for all n, the distance is at least $\text{dist}(C_n) > dn$ and the rate is at least $\text{rate}(C_n) > r$.

We only study linear codes, as they are more mathematically structured and tractable.

Definition 6.5 (Linear Code)**.** A code $C \subset \{0,1\}^n$ is a linear code of dimension k if it forms a k-dimensional vector subspace of $\{0,1\}^n$ over the finite field $\mathbb{F}_2$.

Because a linear code is a vector subspace, it can be defined as the null space (i.e., kernel) of a matrix.

Definition 6.6 (Parity-Check Matrix)**.** A linear code C can be perfectly defined by an $(n-k) \times n$ matrix H_C called the parity-check matrix. A string $x \in \{0,1\}^n$ is a valid codeword if and only if it satisfies all parity constraints in H_C:

$$C = \{x \in \{0,1\}^n \mid H_C x = \mathbf{0} \pmod 2\}.$$

Essentially, each row of H_C represents a single parity constraint, making a specific subset of bits in the codeword to sum to an even number.

For a linear code, the distance $\text{dist}(C)$ is equal to the minimum Hamming weight of any non-zero codeword in C. Also, if a code is defined by an $m \times n$ parity check matrix, it imposes at most m linear constraints. Thus, the dimension of the subspace is at least $n-m$, so the rate of the code is at least $\frac{n-m}{n} = 1 - \frac{m}{n}$.

6.2. **Graph Codes and Tanner Graphs.** The critical insight that links coding theory to expander graphs is representing the parity-check matrix H_C as a bipartite graph. This concept was first pioneered by Robert Gallager in his 1963 MIT doctoral thesis [Gal60].

Given an $m \times n$ parity-check matrix H_C, we construct a bipartite *Tanner graph* $G = (L \cup R, E)$:

- The left vertex set L has n vertices, representing the n bits of the codeword (variable nodes).
- The right vertex set R has m vertices, representing the m rows of the parity-check matrix (constraint nodes).
- An edge connects variable node i to constraint node j if and only if the matrix entry $(H_C)_{j,i} = 1$.

A string $x \in \{0,1\}^n$ is a valid codeword if and only if, for every single constraint node $c \in R$, the sum of the bit values of all variable nodes connected to c is even.

Gallager proposed using very sparse parity-check matrices, giving rise to **Low-Density Parity-Check (LDPC) codes**. Specifically, he considered codes where the underlying Tanner graph is k-left-regular. He proved probabilistically that if one generates a random sparse Tanner graph, it yields an asymptotically good code. However, at the time, explicit expander graphs had not yet been discovered. The field was then inactive until the 1990s, when researchers Michael Sipser and Daniel Spielman revived the area by replacing Gallager's random graphs with explicit expander graphs.

6.3. **Sipser-Spielman Codes.** In 1996, Sipser and Spielman proved that if the bipartite Tanner graph has sufficient vertex expansion, the resulting code is guaranteed to have a minimum distance that scales linearly with n.

We first define the specific type of expansion required for their seminal result.

Definition 6.7 (Left Vertex Expansion Ratio)**.** Let $G = (L \cup R, E)$ be a bipartite graph. Its left vertex expansion ratio $L(G, \delta)$ is defined as

$$L(G, \delta) = \min_{\substack{S \subset L \\ 0 < |S| \leq \delta}} \frac{|\Gamma(S)|}{|S|},$$

where δ is some pre-defined maximum size of the subsets being considered, and $\Gamma(S) \subset R$ denotes the set of all neighbors of S in R.

If the graph is k-left-regular, every vertex in L has exactly k edges. Thus, the absolute maximum possible expansion is $L(G, \delta) = k$. Sipser and Spielman discovered that an expansion greater than $k/2$ will force the existence of "unique neighbors," which prevent low-weight codewords from existing.

Theorem 6.8 (Sipser-Spielman Distance Guarantee (1996))**.** *Let G be a k-left-regular bipartite graph. If the left vertex expansion satisfies $L(G, \delta) > k/2$, then the resulting code $C(G)$ has a minimum distance of at least δ.*

Proof. Let $C(G)$ be the code defined by the Tanner graph G. Since $C(G)$ is a linear code, its minimum distance is exactly equal to the minimum Hamming weight of any non-zero codeword. Suppose for the sake of contradiction that there exists a non-zero codeword $x \in C(G)$ with a Hamming weight w such that $0 < w < \delta$. Let $S \subset L$ be the support of x: S is the set of variable nodes where $x_i = 1$. By our assumption, the size of this set is $|S| = w < \delta$.

Because the expansion of sets up to size δ is strictly greater than $k/2$, we know that the neighborhood of S must satisfy:

$$|\Gamma(S)| > \frac{k}{2}|S|$$

Consider the set of edges originating from S. Because G is k-left-regular, the total number of edges exiting S is exactly $k|S|$. These edges land entirely within the neighborhood set $\Gamma(S)$.

We calculate the average degree of the constraint nodes in $\Gamma(S)$ with respect to these incoming edges from S. The average number of incoming edges per constraint node is

$$\text{Average Degree} = \frac{\text{Total Edges}}{|\Gamma(S)|} = \frac{k|S|}{|\Gamma(S)|} < \frac{k|S|}{k|S|/2} = 2.$$

Because the average degree of the constraint nodes in $\Gamma(S)$ is strictly less than 2, and every constraint node in $\Gamma(S)$ receives at least 1 edge (by definition of being in the neighborhood), there must exist at least one constraint node $c \in \Gamma(S)$ that only receives exactly 1 edge from S. This node c is known as a *unique neighbor*. It is connected to exactly one variable node $v \in S$ that holds the value 1, and to no other variable nodes in S.

Now, consider the parity check equation corresponding to constraint node c. The value

of this parity check is the sum of the bits of its neighbors. Since $x_v = 1$ and all other variable nodes connected to c must belong to $L \setminus S$ (and therefore hold the value 0), the sum evaluated at constraint c is exactly 1 (mod 2). This violates the parity constraint, so x cannot be a valid codeword. This contradicts our initial assumption, proving that no codeword can have a weight strictly less than δ. Thus, $\text{dist}(C(G)) \geq \delta$. ■

By utilizing explicit bipartite expander graphs constructed from combinatorial methods (such as the zig-zag product discussed previously), we can easily satisfy $L(G, \delta) > k/2$ for $\delta = \Omega(n)$. This provides a fully deterministic construction of asymptotically good expander codes with a guaranteed linear distance.

6.4. **Linear-Time Decoding.** While distance and rate are critical, the major advantage of expander codes is their decoding speed. Sipser and Spielman proved that an even stronger expansion threshold of $L(G, \delta) > \frac{3}{4}k$ gives a greedy algorithm can correct a constant fraction of errors in exactly linear $O(n)$ time.

The algorithm operates as such, given a n-bit string y:

(1) Check all m constraint nodes to see if their parity is satisfied or violated.
(2) If there exists any variable node $v \in L$ that is connected to strictly more *violated* constraints than *satisfied* constraints, flip the bit value of v.
(3) Repeat step 2 until no such variable node exists, or all constraints are satisfied.

As a quick note, this algorithm is part of a broader class of *belief propagation* algorithms widely used for decoding of advanced error-correcting codes.

Theorem 6.9 (Sipser-Spielman Efficient Decoder (1996))**.** *Let G be a k-left-regular bipartite graph with left expansion $L(G, \delta) > \frac{3}{4}k$. Let y be an n-bit string whose Hamming distance from a valid codeword x is at most $\delta/2$. Then repeated application of the naive iterative decoder will return the correct codeword x in a linear $O(n)$ number of iterations.*

Proof. Because the code is linear, we can assume without loss of generality that the transmitted codeword was the all-zero string $x = \mathbf{0}$. Therefore, the received string y has exactly $w \leq \delta/2$ ones, which represent the errors. Then a constraint node is violated if and only if it connects to an odd number of 1's.

Let $y^{(i)}$ be the vector generated by the algorithm after i iterations, where $y^{(0)} = y$. Let A_i represent the set of corrupt variable nodes at iteration i, meaning $A_i = \{v \in L \mid y_v^{(i)} = 1\}$. Our goal is to prove that the set A_i becomes empty in linear $O(n)$ iterations i.

Disregarding the index i for simplicity, we assume we are at a specific step where the corrupt set A is non-empty and $|A| \leq \delta$. We now partition the neighborhood $\Gamma(A)$ into two disjoint sets:

- S: The set of *satisfied* neighbors, or constraint nodes connected to an even number of vertices in A.
- U: The set of *unsatisfied* neighbors, or constraint nodes connected to an odd number of vertices in A.

Because $|A| \leq \delta$, we can apply our expansion guarantee:

$$|S| + |U| = |\Gamma(A)| > \frac{3}{4}k|A|. \tag{6.1}$$

Now, we count the total number of edges originating from the corrupt set A. Because G is k-left-regular, there are exactly $k|A|$ such edges. We can bound this total by counting how these edges distribute into U and S. Every node in U receives at least 1 edge from A. Every node in S receives at least 2 edges from A. Thus we have

$$|U| + 2|S| \leq k|A|. \tag{6.2}$$

We linearly combine equations 6.1 and 6.2 to obtain

$$|U| > \frac{1}{2}k|A|, \tag{6.3}$$

which shows that the total number of unsatisfied constraints originating from A is strictly greater than half the total possible maximum edges ($k|A|/2$). By the Pigeonhole Principle, since $|A|$ vertices are sending edges into U, there must exist at least one variable node in A that is connected to strictly more than $k/2$ unsatisfied constraints. Because this node has degree k and more than $k/2$ of its constraints are unsatisfied, it has strictly more unsatisfied neighbors than satisfied neighbors. Thus, the algorithm will flip this bit. When it does so, it will *fix* more than $k/2$ constraints while potentially *breaking* at most $k/2$ constraints. Thus, the absolute number of unsatisfied constraints $|U|$ in the entire graph strictly decreases by at least 1 during every single iteration.

Since the total number of initial constraints is at most $m \leq n$, the number of unsatisfied constraints must hit zero in at most m steps. At this point, the corrupt set A must be empty and the codeword fully corrected.

Finally, we must ensure that the size of the corrupt set A never accidentally exceeds the expansion boundary δ while the algorithm runs. Initially, $|A_0| \leq \delta/2$. At every step, the size of A changes by exactly ± 1. If $|A|$ ever attempted to cross the threshold δ, there would exist an intermediate step i where $|A_i| = \delta$. But by our previous equations, this would imply $|U_i| > k\delta/2$. However, we know $|U|$ is strictly decreasing at every step. In the very beginning, the maximum number of unsatisfied constraints was bounded by the total edges leaving A_0, which is $k|A_0| \leq k\delta/2$. This creates a contradiction, so the corrupt set can never increase past δ. Thus the expansion bound always holds and the algorithm converges in $O(n)$ time. ■

7. Applications II: Random Walks on Expander Graphs

While the combinatorial and algebraic properties of expander graphs allow for the construction of robust networks and error-correcting codes, their probabilistic properties—via random walks—are as, if not more, influential. In theoretical computer science, randomness in algorithms leads to unpredictability and decreased reproducibility. Expander graphs serve as a tool for *derandomization*: reducing the error of a random algorithm without significantly increasing the random bits needed (especially useful for space-bounded computation). We will explore how a random walk on an expander graph can exponentially reduce algorithmic error with only a few more random bits.

7.1. **Mixing Time.**

A random walk on a graph G begins at some initial probability distribution π over the vertices. At each step, the walk moves to a uniformly chosen neighbor of the current vertex. For a non-bipartite, connected d-regular graph, this discrete-time Markov chain is guaranteed to converge to the uniform stationary distribution $u = (\frac{1}{n}, \ldots, \frac{1}{n})^T$. The

mixing time is defined as the number of steps required for the distribution of the walk to become extremely close to u. For general regular graphs, mixing can take polynomial time (e.g., $O(n^2 \log n)$). For expander graphs, mixing happens exponentially faster.

Theorem 7.1 (Mixing Time of Expanders)**.** *Let G be a connected, non-bipartite d-regular graph on n vertices with normalized second eigenvalue magnitude $\alpha < 1$. The mixing time to reach a distribution $\pi^{(t)}$ such that $\|\pi^{(t)} - u\|_\infty < \frac{1}{2n}$ is $O\left(\frac{\log n}{1-\alpha}\right)$.*

Because an expander family maintains a constant spectral gap bounded away from 1, $1 - \alpha$ is a constant. Thus, the mixing time on an expander graph is $O(\log n)$.

This rapid mixing of expander graphs has applications in certain algorithms. The classic Undirected $s - t$ Connectivity problem asks whether a path exists between two vertices s and t. A simple randomized algorithm takes a random walk starting at s. If the graph is an expander, the walk will visit almost every vertex in a spectacularly short amount of time, solving connectivity using barely any memory (logarithmic space to store the current vertex). This insight was also the foundation Omer Reingold utilized in 2004 to prove that deterministic logarithmic space equals symmetric logarithmic space ($\mathbf{L} = \mathbf{SL}$).

7.2. **Hitting Property.** The core theorem underlying derandomization is the *hitting property* of expander graphs originally formulated by Ajtai, Komlós, and Szemerédi in 1987. It answers the question: If we define a "bad" subset of vertices B, what is the probability that a random walk gets trapped inside B for k consecutive steps? If the vertices were chosen completely independently at random, this probability would be μ^k, where $\mu = |B|/|V|$. Taking a random walk on an expander, as it turns out, gives a result very similar to this.

Theorem 7.2 (Hitting Property of Expander Graph Random Walks)**.** *Let $G = (V, E)$ be a d-regular graph with normalized spectral expansion α. Let $B \subset V$ be any subset of vertices with density $\mu = |B|/|V|$. If we choose a starting vertex v_0 uniformly at random and take a k-step random walk $v_0, v_1, \ldots, v_k$, the probability that the entire walk remains inside B is bounded by:*

$$\Pr[v_0, \ldots, v_k \in B] \leq (\mu + \alpha)^k$$

In fact, more complex spectral analysis, as demonstrated by Impagliazzo and Zuckerman, refines this upper bound tightly to $(\mu^2 + \alpha^2)^{k/2}$, although the theorem presented above is sufficient for asymptotic derandomization.

7.3. **Randomness-Efficient Error Reduction.** The most famous application of the hitting property is error reduction of randomized algorithms without significantly increasing the number of random bits.

Consider a problem in the complexity class **RP** (randomized and polynomial time). An algorithm $\mathcal{A}$ to solve this problem uses r random bits to verify if an input x belongs to a language L:

- If $x \in L$, $\mathcal{A}$ outputs "YES" with probability $\geq 1/2$.
- If $x \notin L$, $\mathcal{A}$ outputs "NO" with probability 1.

The algorithm has a **one-sided error**: if it outputs "YES", it is definitely correct. But if it outputs "NO", it might have simply experienced bad luck with its random bits. Algorithms with **two-sided error** belong to the complexity class BPP (bounded-error, probabilistic,

and polynomial time), and error reduction algorithms exist for that.

To reduce the failure probability from $1/2$ to a negligible 2^{-k}, the naive approach is to run the algorithm k independent times. If any run outputs "YES", we accept. The probability of failing k times is $(1/2)^k = 2^{-k}$. However, this naive approach requires $k \cdot r$ purely independent random bits. Since generating true randomness is computationally expensive, this is highly inefficient.

In 1989, Impagliazzo and Zuckerman demonstrated how to achieve the exact same exponential error drop using only $r + O(k)$ random bits via expander graphs. Their algorithm was as follows:

(1) Construct an explicit d-regular expander graph G where the vertex set is $\{0,1\}^r$ (all possible random seeds for $\mathcal{A}$). Ensure the spectral expansion α is small.
(2) Pick a starting vertex v_0 uniformly at random. This costs r random bits.
(3) Take a random walk of length k: $v_0, v_1, \ldots, v_k$. At each step, there are d neighbors, so choosing the next step requires only $\log_2(d)$ random bits. This costs $r + k\log_2(d) = r + O(k)$ random bits.
(4) Run the algorithm $\mathcal{A}$ using $v_0, v_1, \ldots, v_k$ as the random seeds. If *any* v_i causes $\mathcal{A}$ to output "YES", accept (i.e., take an OR operation).

We may analyze this algorithm.

For a valid input $x \in L$, the algorithm only fails if *every single seed* v_i gives a "NO". Let B be the set of "bad" random strings that produce a "NO". Because the algorithm's base success rate is $1/2$, the density of the bad set is $\mu = |B|/2^r$. The probability that the algorithm fails is exactly the probability that the entire random walk $v_0, \ldots, v_k$ gets trapped inside the bad set B. By the Hitting Property theorem presented previously, this probability is bounded by

$$\Pr[\text{Failure}] \leq (\mu + \alpha)^k.$$

For sufficient $\mu + \alpha < 1/4$, this probability becomes less than 4^{-k}. This is already an exponential reduction! With better bounds for the hitting property, stronger expanders (i.e., decreased α), and some variations on the algorithm (e.g., taking a majority of the vertices instead of an OR operation), the error can be even further driven down. Thus we have achieved an exponential reduction in error by adding only a constant number of random bits per step.

7.4. **The PCP Theorem.** A related application of random walks on expander graphs is in the proof of the PCP Theorem. PCPs are Probabilistically Checkable Proofs, and the PCP theorem is arguably the most significant result in computational complexity theory. It states that every problem in **NP** has a proof that can be verified by only examining a constant number of bits from the proof using only a logarithmic number of random bits. While algebraic proofs exist, Irit Dinur presented a clean combinatorial proof in 2007 based on expander graph techniques. She essentially performed random walks on expander graphs to amplify the "distance" between valid and invalid proofs, allowing a verifier to reject an invalid proof with high probability by checking only a constant-sized portion of the proof structure. Active areas of research today extending PCPs include interactive proofs and Quantum PCPs.

8. Further Reading and Current Research

For a comprehensive, graduate-level survey of expander graphs, see the seminal survey by Hoory, Linial, and Wigderson [HLW06]. To further explore random walks and derandomization, see Vadhan's extensive work on pseudorandomness [Vad12]. Trevisan's lecture notes [Tre16] and Urqidi's master's thesis [Urq10] also provide excellent surveys of expander graphs and applications. For more on error-correcting codes, see Guruswami's work [Gur04].

Current research on expander graphs has largely shifted to higher-order generalizations, quantum extensions, and advanced combinatorial constructions.

8.1. **High-Dimensional Expanders.** In the last decade, the theory of "high-dimensional expanders" (HDXs) has begun to emerge. While classical expanders are graphs (which are one-dimensional simplicial complexes) that are simultaneously sparse and highly connected, high-dimensional expanders generalize this framework to finite simplicial complexes of dimension $d \geq 2$. Generalizing expander properties to higher dimensions has proven more difficult than thought. Thus mathematicians have created many more different types of expanders, including coboundary expanders, cosystolic expanders, topological expanders, geometric expanders, and spectral expanders [Lub17].

The earliest work related to HDXs is considered to be Garland's 1973 proof about the cohomology groups of arithmetic lattices of p-adic Lie groups. More recently, topological expanders have been explicitly constructed from finite quotients of Bruhat-Tits buildings. These structures are currently being explored for their applications in topological data analysis [Lub17].

8.2. **Quantum Expander Codes.** Another large area of modern research lies in **quantum** error correction. Researchers are currently exploring quantum low-density parity-check (qLDPC) codes as a way to remove noise from quantum information.

One highly studied class of qLDPC codes is hypergraph product codes, which possess a constant rate. These codes utilize a linear-time decoder known as small-set-flip (SSF), but SSF displays sub-optimal performance in practice and requires massive code sizes to be effective. Recent breakthroughs have successfully constructed hybrid decoders that combine the classical belief propagation algorithm with the SSF decoder, significantly improving error thresholds. These low-complexity, high-performance decoders provide evidence that constant-rate qLDPC codes are worth examination for large-scale quantum computation [GGKL21].

8.3. **Combinatorial Constructions: Lifts.** Beyond algebraic structures and zig-zag products, recent deterministic constructions of expanders often utilize the "lift" operation. Literature also explores natural extensions of this operation, such as the "partial lift," which can be used to "interpolate" between graphs within a known expander family. This interpolation produces intermediate graphs maintaining strong spectral expansion properties with minimal structural differences from their parent graphs [Kol24].

References

[AKS83] M. Ajtai, J. Komlos, and E. Szemeredi. O(n log n) sorting network. Conference Proceedings of the Annual ACM Symposium on Theory of Computing, pages 1–9. ACM, 1983.

[AKS87] Miklós Ajtai, János Komlós, and Endre Szemerédi. Deterministic simulation in logspace. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 132–140, 1987.

[Din07] Irit Dinur. The pcp theorem by gap amplification. *Journal of the ACM (JACM)*, 54(3):12–es, 2007.

[Gal60] Robert G Gallager. Low density parity check codes, 1960.

[GG81] Ofer Gabber and Zvi Galil. Explicit constructions of linear-sized superconcentrators. *Journal of Computer and System Sciences*, 22(3):407–420, 1981.

[GGKL21] Antoine Grospellier, Lucien Grouès, Anirudh Krishna, and Anthony Leverrier. Combining hard and soft decoders for hypergraph product codes. *Quantum*, 5:432, April 2021.

[Gur04] Venkatesan Guruswami. Guest column: error-correcting codes and expander graphs. *ACM SIGACT News*, 35(3):25–41, 2004.

[HLW06] Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.

[INW94] Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 356–364, 1994.

[IZ89] R. Impagliazzo and D. Zuckerman. How to recycle random bits. In *30th Annual Symposium on Foundations of Computer Science*, pages 248–253, 1989.

[Kol24] Josh Kolenbrander. Expander graphs, lifts, and applications. 2024.

[LPS88] Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988.

[Lub17] Alexander Lubotzky. High dimensional expanders, 2017.

[Mar73] Grigorii Aleksandrovich Margulis. Explicit constructions of concentrators. *Problemy Peredachi Informatsii*, 9(4):71–80, 1973.

[Nil91] Alon Nilli. On the second eigenvalue of a graph. *Discrete Mathematics*, 91(2):207–210, 1991.

[PF73] M. Pinsker and Ferro Fn. On the complexity of a concentrator. 1973.

[Rei08] Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4), September 2008.

[RVW02] Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders. *Annals of mathematics*, pages 157–187, 2002.

[SS96] Michael Sipser and Daniel Spielman. Expander codes. *IEEE Tkans. Inform. Theory*, 42:1710–1722, 1996.

[Tan81] R. Tanner. A recursive approach to low complexity codes. *IEEE Transactions on Information Theory*, 27(5):533–547, 1981.

[Tre16] Luca Trevisan. Lecture notes on graph partitioning, expanders and spectral methods. University of California, Berkeley, 2016. Course CS294.

[Urq10] José Miguel Pérez Urquidi. Expander graphs and error correcting codes. Master thesis, U.F.R. Mathématiques et Informatique, 2010.

[Vad12] Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1–3):1–336, December 2012.

[Zem01] G. Zemor. On expander codes. *IEEE Transactions on Information Theory*, 47(2):835–837, 2001.

Appendix A. Proof of Discrete Cheeger Inequality

We provide a proof for the Discrete Cheeger Inequality (Theorem 4.1) in two parts: the lower bound and the upper bound. Both require heavy linear algebra.

Lemma A.1 (Spectral Lower Bound on Expansion)**.** *If G is a d-regular graph, then $\frac{d-\lambda_2}{2} \leq h(G)$.*

Proof. Let $S \subset V$ be the subset that perfectly achieves the minimum edge expansion ratio, meaning $h(G) = \frac{|\partial S|}{|S|}$ and $|S| \leq \frac{n}{2}$. We define the vector $w \in \mathbb{R}^n$ using $\mathbf{1}_S$ and $\mathbf{1}_{\overline{S}}$, the characteristic indicator vectors of the sets S and $\overline{S}$ respectively, as:

$$w = |\overline{S}|\mathbf{1}_S - |S|\mathbf{1}_{\overline{S}}.$$

First, w is orthogonal to the all-ones vector $\mathbf{1}$ because the sum of its coordinates is $|\overline{S}||S| - |S||\overline{S}| = 0$. Thus, because w resides entirely in the subspace orthogonal to the primary eigenvector v_1, applying the Rayleigh quotient guarantees that $wAw^T \leq \lambda_2 \|w\|^2$. We calculate the squared Euclidean norm of w:

$$\|w\|^2 = \sum_{i=1}^{n} w_i^2 = |S||\overline{S}|^2 + |\overline{S}||S|^2 = |S||\overline{S}|(|S| + |\overline{S}|) = n|S||\overline{S}|.$$

Next, we expand wAw^T:

$$wAw^T = 2|E(S)||\overline{S}|^2 + 2|E(\overline{S})||S|^2 - 2|\partial S||S||\overline{S}|$$

By the Handshaking Lemma, we have $2|E(S)| = d|S| - |\partial S|$ and $2|E(\overline{S})| = d|\overline{S}| - |\partial S|$. Substituting these identities into our quadratic form and expanding terms algebraically, we have

$$wAw^T = nd|S||\overline{S}| - n^2|\partial S|.$$

Substituting back into the Rayleigh quotient inequality, we obtain:

$$nd|S||\overline{S}| - n^2|\partial S| \leq \lambda_2 n|S||\overline{S}|$$

Dividing both sides by $n|S||\overline{S}|$ and rearranging terms yields:

$$d - \lambda_2 \leq \frac{n}{|\overline{S}|}\frac{|\partial S|}{|S|}$$

By our initial assumption, S contains at most half the vertices, meaning $|S| \leq \frac{n}{2}$ and consequently $|\overline{S}| \geq \frac{n}{2}$. Therefore, the fraction $\frac{n}{|\overline{S}|}$ is bounded above by 2. We substitute this bound and recall that $\frac{|\partial S|}{|S|} = h(G)$:

$$d - \lambda_2 \leq 2h(G) \implies \frac{d - \lambda_2}{2} \leq h(G),$$

completing our proof. ■

Lemma A.2 (Spectral Upper Bound on Expansion)**.** *If G is a d-regular graph, then $h(G) \leq \sqrt{2d(d - \lambda_2)}$.*

Proof. Let $L = dI - A$ be the unnormalized Laplacian matrix of G. The eigenvalues of L are $d - \lambda_i$. Let g be the eigenvector corresponding to the second smallest Laplacian eigenvalue $d - \lambda_2$. Because g is orthogonal to the constant all-ones vector $\mathbf{1}$, its components sum to zero, meaning g must possess both positive and negative entries. Define $f = g^+$ as the positive part of g, such that $f_v = \max(g_v, 0)$ for all $v \in V$. Let $V^+ = \{v \in V \mid f_v > 0\}$ be the support of f. By negating the eigenvector g if necessary, we can guarantee that $|V^+| \leq \frac{n}{2}$. Evaluating the x-th component of Lf for any $x \in V^+$:

$$(Lf)_x = df_x - \sum_{y \in V} a_{xy} f_y = dg_x - \sum_{y \in V} a_{xy} g_y^+.$$

Because $g_y \leq g_y^+$ for all y, it follows that $-g_y^+ \leq -g_y$. Substituting this yields $(Lf)_x \leq (Lg)_x$. Since g is an eigenvector of L, we also have

$$(Lg)_x = (d - \lambda_2)g_x = (d - \lambda_2)f_x.$$

Therefore, evaluating the Rayleigh quotient on f:

$$fLf^T = \sum_{x \in V^+} f_x(Lf)_x \leq \sum_{x \in V^+} f_x(d - \lambda_2)f_x = (d - \lambda_2)\|f\|^2.$$

We now relate this Rayleigh quotient to the graph. By applying the Cauchy-Schwarz inequality over the edges E, we bound the sum of differences of squares:

$$\sum_{\{x,y\} \in E} |f_x^2 - f_y^2| = \sum_{\{x,y\} \in E} |f_x - f_y|(f_x + f_y) \leq \sqrt{\sum_{\{x,y\} \in E} (f_x - f_y)^2} \sqrt{\sum_{\{x,y\} \in E} (f_x + f_y)^2}.$$

On the RHS, the first term under the square root is exactly fLf^T. For the second term, using the inequality $(a+b)^2 \leq 2(a^2+b^2)$ and the d-regularity of G, we find $\sum (f_x + f_y)^2 \leq 2d\|f\|^2$. This implies:

$$\sum_{\{x,y\} \in E} |f_x^2 - f_y^2| \leq \sqrt{fLf^T} \sqrt{2d\|f\|^2}.$$

The left hand side is exactly equal to $\int_0^\infty |\partial S_t| \, dt$, where $S_t = \{x \in V \mid f_x^2 > t\}$. Because the support of f is at most $\frac{n}{2}$, every level set S_t satisfies $|S_t| \leq \frac{n}{2}$. Therefore, $|\partial S_t| \geq h(G)|S_t|$. We substitute this into the integral:

$$\int_0^\infty |\partial S_t| \, dt \geq h(G) \int_0^\infty |S_t| \, dt = h(G)\|f\|^2.$$

Combining our inequalities, we obtain $h(G)\|f\|^2 \leq \sqrt{fLf^T}\sqrt{2d\|f\|^2}$. Squaring both sides and substituting our earlier bound $fLf^T \leq (d - \lambda_2)\|f\|^2$ yields:

$$h(G)^2 \leq 2d \frac{fLf^T}{\|f\|^2} \leq 2d(d - \lambda_2).$$

Taking the square root completes the proof. ■