Université de Bordeaux 1
Sciences et Technologies

U.F.R. Mathématiques et Informatique

Master Thesis

Expander Graphs and Error Correcting Codes

Thesis Advisor	**Candidate**
Prof. Chistine Bachoc	José Miguel Pérez Urquidi

Academic Year 2009–2010

Acknowledgments

I am deeply grateful to the people without whom
this work would not have been possible,

to my supervisor, Prof. Christine Bachoc,
for all her time and patience while helping me write this thesis,

to my family, who has always supported
and encouraged me at all times,

to all my friends, for their kind words and shared moments.

Miguel Urquidi

Contents

Introduction

When asked to list our day to day activities one may mention going to work, practicing a sport, buying groceries or drinking a cup of coffee with a friend, among many others. It seems that we fail to include “talking to people” in the list even though we probably do so as a part of each of the listed activities. Talking to other people is an everyday activity so frequent and common that we tend to overlook it, and we may not realize its complexity. Essentially, it is transmitting a message between two individuals, from a speaker to a listener. Although we talk using words, the messages that we want to transmit are actually ideas. An idea is difficult to transmit as it is, that is why an acceptable sequence of words is chosen to represent the idea, which we call the encoding of the message. Note that the choice of the encoding is not unique, there are different sentences that could express the same idea, or we could decide to encode the same idea in a different language, sometimes we might even resort to screaming instead of words to express pain. It should be clear from the previous examples that different encodings are not equivalent, and that some might prove more effective or efficient than others. Upon reception of the encoded message, it is the turn of the listener to implement the inverse routine, which is called the decoding, where we transform the words back to ideas and we are able to understand each other.

The transmission of messages does not pertain only to the speaker and listener, it is also important the channel through which one is communicating, because if the channel is noisy it can produce some errors. This noisy channel can be thought of as the interference produced by the noise from cars, from a crowd, or by speaking over the phone. A simple solution to this problem is to ask the speaker to repeat what he said, and hopefully we will understand the second or third time he repeats it. However, a more efficient approach is to correct the error ourselves, which we do in daily life, probably without realizing it.

All the ideas expressed above are intuitively the field of study of Coding Theory, a rich and active field initiated by Shannon's work. To solve the noisy channel problem, he suggested creating a dictionary or code of acceptable words to be transmitted, and a sense of distance between the words, not far at all from the concept of a real life dictionary. Then upon reception of an encoded message, if it is in the dictionary it is accepted, otherwise it has been corrupted by the channel and we try to find the closest word to it in the dictionary. This is called an error-correcting code. The principle goal of this work, is to present constructions of good error-correcting codes.

I Cdnuolt blveiee taht I cluod aulaclty uesdnatnrd waht I was rdanieg.

That was another example of an error-correcting code, our mind checks for the closest possible words, we do this automatically and it enables us to read the previous sentence. All along we have been talking about transmitting messages, which are ideas by the use of words, but the messages could be anything, their nature is arbitrary. This is one of the reasons why error-correcting codes are important, because they can be used in numerous different situations. Just to mention a few examples they are implemented to overcome the lost of data by scratches when reading a CD, to compensate for interferences over long distances in cell phone calls, to read barcodes of products, and in all communications between computers over the internet. Once we embrace the fact that the nature of the messages is arbitrary, we can find error-correcting codes almost everywhere.

A dictionary has two conflicting desired properties, the distance between its words and the amount of words in it, the first allows to correct errors, while the second makes the code more efficient. As mentioned before, our goal is to find good error-correcting codes, and good codes posses a compromise between the two properties. We are in fact interested in finding not single good codes but rather families of good codes, better explained as asymptotically good codes in the last chapter. One can relate a code to linear algebra by a matrix, called the parity check matrix of the code.

A graph is roughly speaking, a set of points of vertices together with a set of lines or edges that join pairs of vertices. As messages in coding theory, the nature of the vertices and edges of a graph is arbitrary, which renders them very versatile, among others fields they are present in mathematics, social sciences, computer science, and statistical physics.

Suppose, for example, that in a graph the vertices represent computers while edges between them represent that the computers are connected to each other. We would like the computers to be well connected amongst them, this is the notion of expansion of a graph, which can be measured in many different ways, one of which is counting outgoing connections of small subsets. If this was all we could simply consider every computer to be connected to every other computer, providing a good expansion for the associated graph. However, if the wiring to establishing the connections between computers has a high price, we would like to achieve expansion without too many connections. Based on these observations, and wondering if constructions of graphs with good expansion can be given for any number of computers, one arrives to the definition of a family of expander graphs. A family of expander graphs has two conflicting desired properties, it should be well connected which requires edges, but at the same time it should have sparse edges. One can also associate a matrix to a graph, which is called the adjacency matrix of the graph. Studying the eigenvalues of the matrix one can infer properties about the graph, particularly about its expansion.

Using their associated matrices, we can link the two fields of Coding Theory and Graph Theory. Furthermore, a proper construction we can match the conflicting properties of a graph to those of a code, thus finding a good graph leads us to a good code, in other words, from an explicit family of expander graphs we can construct explicit error-correcting codes. We provide two distinct constructions of asymptotically good error correcting codes. The first one utilizes the link between the fields in a very straightforward fashion, considering the adjacency matrix of certains graphs as the parity check matrix of the code. The second construction uses a family of expander graphs and a fixed small code to construct a larger code for each graph in the family, thus providing asymptotically good error-correcting codes. This construction was further improved by Gilles Zémor by using a special kind of graph in the family of expander graphs, the Ramanujan graphs.

From what has been exposed previously we divide the thesis in 6 chapters. We give a brief summary of what is done in each one of them.

- Chapter 1. The objective of this chapter is to introduce the reader to most of the definitions of Graph Theory that will be used in subsequent chapters.

  We start by giving the basic definitions of the different types of graphs, such as directed graph, undirected graph, simple graph, multigraph, finite graph and infinite graph.

  We then introduce several concepts on the field, amongst other paths and cycles in a graph, neighbors of a vertex, isolated vertices, the degree of a vertex and degree of a graph. We also introduce some useful notation for the set of edges between sets, the neighbors of a vertex and neighbors of a set.

  Important and common families of graphs are also mentioned, for example the complete graphs, bipartite graphs and complete bipartite graphs.

- Chapter 2. The main purpose of this chapter is to provide the link between Graph Theory and Linear Algebra, by defining the associated matrices to a graph G, particularly its adjacency matrix.

  First, we define formally the adjacency matrix of a graph G. Then we consider its eigenvalues and define them to be its spectrum. We provide a few minor results involving the eigenvalues, and we prove the Expander Mixing Lemma.

  Then we continue to define other matrices associated to a graph G, such as its gradient, its divergence and its Laplacian.

- Chapter 3. Here, our goal is to establish the concept of expansion and derive upper and lower bounds for it.

  Again, we introduce the new concepts, which are the expansion $h(G)$ of a graph G, and a family of expander graphs.

We then proceed to prove the bounds on the expansion $\frac{d-\lambda_2}{2} \leqslant h(G) \leqslant \sqrt{2d\,(d-\lambda_2)}$. This shows clearly that there is strong relation between the eigenvalues of the adjacency matrix of a graph and its expansion.

- Chapter 4. This section is dedicated to introduce the constructions of new graphs from previous graphs.

  This is a rather short chapter, introducing the replacement product of two graphs, the regular zig-zag product and the zig-zag product for bipartite graphs, which can be used to build families of expander graphs.

- Chapter 5. Finally in this chapter, we provide an explicit construction of expander graphs, which are due to Margulis and bear the same name.

  We start by giving an infinite analog of the Margulis construction. Then we define formally the family of Margulis graphs. Finally we prove that given family of graphs is indeed a family of expander graphs.

- Chapter 6. Before we give the constructions of error correcting codes, we must first introduce Coding Theory and related definitions and concepts. We explain the noisy channel problem and the solution suggested by Shannon of using a dictionary of codewords.

  We define the Hamming distance between vectors, along with the distance of a code and its rate. Then, we give a formal definition of what we are after, a family of asymptotically good codes.

  Next, we introduce linear codes and its link to Linear Algebra through the parity check matrix of a code C. Directly followed by proving some bounds on the distance and rate of a code, namely the Gilbert-Varshamov and the Sphere-Packing bounds.

- Chapter 7. In this chapter we yield two different constructions of asymptotically good and efficient linear codes, based on the explicit constructions of expander graphs.

  At this point we explain the first, which uses bipartite expander graphs. The decoding uses a called belief propagation algorithm, which based on the neighbors of a vertex x of the graph, it flips its related bit in the code if it has more unsatisfied than satisfied neighbors.

  To conclude we provide the second construction, which was presented by Sipser and Spielman using regular expander graphs, then later improved by Gilles Zémor by considering the special case of Ramanujan graphs and using a small variant of the decoding algorithm. The construction uses both bipartite Ramanujan graphs and a small linear code to build a larger linear code. The decoding algorithm consists of applying complete decoding of the smaller linear code induced on the vertices of the Ramanujan graph, alternating between its left and right sets in each iteration.

Chapter 1

Graph Theory

The objective of this chapter is to provide an introduction to graph theory, establishing most of the concepts and vocabulary that will be used in the following chapters, it consists mainly in definitions in the field. Only the basics will be covered here, more definitions and results will be introduced in later chapters as they are needed.

Let it be clear that, although it is a vast and interesting area, this work is not focused in graph theory, rather in it versatility to be applied to other fields, in our case to build efficient error correcting codes. The reader may consult the books [1, 3] for more detailed explanations of the concepts introduced in this chapter, or to further read on the subject.

1.1 Basic Definitions

A graph is intuitively a set of points and set of lines, each of which joins a pair of points. The points are referred to as vertices, and the lines are also known as edges. The nature of the vertices and the edges is arbitrary, this is what gives graphs such versatility. For example, the vertices could represent men or women, and then the edges the relationships between them; or the vertices could be computers, while the edges signify that two computers can communicate, which would be a simple representation of the internet. In our case, vertices may be variables or constraints in a system, and edges indicating which variables are susceptible to which constraints, which can be used to construct a code.

We proceed now to give the formal definition of a graph.

Definition 1.1.

*A **graph** G is an ordered pair $(V(G), E(G))$, which consists of the disjoint sets of **vertices** $V(G)$ and **edges** $E(G)$, together with an **incidence function** ψ_G associating each edge with an unordered pair of not necessarily distinct vertices. When the context is clear, the sets are simply denoted $V = V(G)$ and $E = E(G)$.*

The form the graph has been defined is particularly known as **undirected graph**, if the incidence function associates instead ordered pairs of vertices then the graph would be called **directed graph**.

The definition is very general, it allows loops and multiple edges. A **loop** is and edge associated to a pair $\{v, v\}$ of vertices, and if several edges are associated to the same pair of vertices such vertices are said to have **multiple edges**. A graph allowing this is commonly known as **multigraph**, and it is said to be a **simple graph** if there are no loops or multiple edges.

This definition also allows for **infinite graphs** to take place, in which the cardinality of the set of vertices or edges is infinite, if not it is a **finite graph**. We will usually consider finite graphs unless otherwise specified.

As we will more often consider simple finite graphs in this work, it is perhaps better to think of a more simple definition for them.

Definition 1.2.

An ***undirected simple (finite) graph*** $G = (V, E)$ *consists of a finite set* V *of vertices together with a set* $E \subset V^2$ *of edges, where* E *is a set of unordered pairs of distinct vertices.*

Remark 1.3.

Note that a simple undirected graph G *can be considered the equivalent of a multi-edge directed graph* H*, where* $V(G) = V(H)$*, and an edge* $\{u, v\} \in E(G)$ *if and only if we have both* $(u, v), (v, u) \in E(H)$*.*

We might use sometimes the shorter notation uv for an undirected edge. Additionally, for a directed edge (u, v), we say that the edge **exits** or **starts** at u, and that it **enters** or **ends** at v. Taking in consideration Remark 1.3, in an undirected graph we can consider that edges both exit and enter a given vertex u. Both in a directed and an undirected graph, for an edge $e = (u, v)$, it is said that the edge is incident to u and v, that u and v are the endpoints of the edge e, and that u and v are **adjacent** to each other.

1.2 Attributes of a Graph

Given two distinct vertices $u, v \in V$, a **path** from u to v is a sequence of distinct vertices $u, w_1, \ldots, w_{n-1}, v$ such that from each vertex there is an edge connecting it to the next one in the sequence, and the path is said to be of length n. If the given vertices u, v are equal, then the sequence of vertices is called instead a **cycle**. Intuitively, we can walk on the edges from u to v. Now we can introduce the notion of connectedness.

Definition 1.4.

In a graph G, two vertices u and v are said to be ***connected*** *if there is a path from u to v, otherwise, they are* ***disconnected****. We say that G is a* ***connected graph*** *if every pair of distinct vertices are connected, otherwise, it is a* ***disconnected graph****.*

A vertex is considered **isolated** if it is not connected to any other vertex in the graph. This is related to the number of edges associated to the vertex, which we define now as its degree.

Definition 1.5.

The ***degree of a vertex*** *v is the number of edges incident to the vertex, with each loop counted twice, and it is denoted* $\deg(v)$*. The* ***degree of a graph*** *$G = (V, E)$ is defined as*

$$\Delta(G) = \max_{v \in V} \deg(v).$$

Therefore, an isolated vertex is a vertex of degree 0. A graph G such that all its vertices have the same degree is called a **regular graph**, and if all its vertices have the same degree d then it is called a **d-regular graph** or regular graph of degree d. At last, a **(n,d)-graph** is a d-regular graph with n vertices.

The average degree of the vertices of a graph $G = (V, E)$ is denoted $\overline{\deg}_G$ or simply $\overline{\deg}$ if the context is clear, thus, explicitly we have

$$\overline{\deg}_G \cdot |G| = \sum_{v \in G} \deg(v).$$

A **subgraph** G' of a graph $G = (V, E)$ is a subset of the vertices $V' \subset V$, along with a subset of the edges $E' \subset E$ such that for all $uv \in E'$ we have $u, v \in V'$. For any given subset S of the vertices V of a graph G, we define the **induced subgraph** G_S of G as the graph whose vertex set is $V(G_S) = S$, and whose edge set consists of all edges in G whose endpoints are contained in S, i.e. the edge set is $E(G_S) = \{uv \in E : u, v \in S\}$.

For a vertex v, the vertices to which it is associated by an edge are called its neighbors.

Definition 1.6.

*The **neighbors of a vertex** v of a graph $G = (V, E)$ are defined as*

$$\Gamma(v) = \{u \in V \ : \ uv \in E\}.$$

*Analogously, for a subset $S \subset V$, we define the **neighbors of the set** as*

$$\Gamma(S) = \{u \in V \setminus S \ : \ \exists v \in S \text{ such that } uv \in E\}.$$

Notice that in a simple graph, the number of neighbors of a vertex v is equal to its degree, i.e. $\Gamma(v) = \deg(v)$. This notation for neighbors gives us a way to manage the vertices associated to a vertex, to manage the edges between sets we introduce the following.

Definition 1.7.

In a graph $G = (V, E)$, for subsets of vertices $S, T \subset V$, we denote the set of edges from S to T by

$$E(S, T) = \{uv \in E \ : \ u \in S, v \in T\}.$$

*The **edge boundary** of a subset S of vertices is $\partial S = E(S, \overline{S})$.*

Particularly, for a subset $S \subset V$ we denote the set $E(S, S)$ simply as $E(S)$. Also note that in an undirected graph $\partial S = \partial \overline{S}$. Observe that with this notation the induced subgraph G_S of a graph G has edge set $E(G_S) = E(G) \cap E(S)$. Finally, we remark that analogously, the neighbors $\Gamma(S)$ of a subset S can be thought of as the vertex boundary of the subset.

We will now establish the concepts of diameter and radius of a graph, but first we must define a sense of distance. The **distance** $d(u, v)$ between two vertices u, v of a graph G is the length of a shortest path joining them, if there is no such path then we define $d(u, v) = \infty$.

Definition 1.8.

The ***diameter*** $\mathrm{diam}(G)$ *of a graph* $G = (V, E)$ *is the longest distance between any two vertices in it,*

$$\mathrm{diam}(G) = \max_{u,v \in V} d(u, v).$$

While the ***radius*** $\mathrm{rad}(G)$ *of the graph is the minimum distance at which all vertices may reach a common vertex,*

$$\mathrm{rad}(G) = \min_{u \in V} \max_{v \in G} d(u, v).$$

We conclude this chapter by introducing the concept of automorphism of a graph.

Definition 1.9.

An ***automorphism*** *of a graph* $G = (V, E)$ *is a permutation* π *of the vertices such that* $uv \in E$ *if and only if* $\pi(u)\pi(v) \in E$.

The automorphisms of a graph reflect its symmetries, if an automorphism π sends u to v, then these vertices are similar. A graph in which all vertices are similar is called **vertex-transitive**, note that in such a graph all permutations of the vertices define an automorphism. On the other hand, if there are no two similar vertices the graph is called **asymmetric**, and it has only the identity as an automorphism. We remark that the set of automorphisms of G forms a group.

1.3 Graphs Classes

In this section we briefly describe some common classes or families of graphs, which we will often refer to in coming chapters. Actually, we have already mentioned some classes, such as the class of regular graphs. Now, consider $G = (V, E)$ to be a graph with $|V| = n$ for the following definitions.

A graph G is called a **path graph**, if there is a sequence of all its vertices forming a path, and if so it is denoted as $G = P_n$.

Analogously, G is a **cycle graph**, denoted C_n, if all its vertices form a cycle.

The graph G is a **complete graph**, denoted K_n, if for every $u, v \in V$ there exists an edge connecting them. Notice that it is a (n-1)-regular graph.

If the set of vertices V can be decomposed into two disjoint sets A and B, such that for every $uv \in E$ we have $u \in A$ and $v \in B$ (or vice versa), then G is known to be a **bipartite graph**, and it is denoted as $G(A, B, E)$. The set A is called the left vertex set, and B the right vertex set. Additionally, if all vertices in A have the same degree d, then G is said to be a d**-left-regular graph.**

A particular case of bipartite graphs, if $|V| = n + m$, $|A| = n$, $|B| = m$, and for every $u \in A$ and $v \in B$ we have that $uv \in E$, then G is a **complete bipartite graph**, denoted $K_{n,m}$. Notice that in this case G is a $|B|$-left-regular graph.

The n**-cube** Q_n is the graph whose vertex set is the set of all n-tuples of 0's and 1's, where there is and edge between two n-tuples if they differ in precisely one coordinate. We remark that the n-cube has 2^n vertices, and it is a n-regular graph.

There is an important family of graphs, which will become vital for us in one of the constructions of error correcting codes, we are talking about the family of Ramanujan graphs. However, at this point we are not in no position to even define them, in the next chapter we will introduce a link between graphs and linear algebra which, among other benefits, will allow us to define a Ramanujan graph.

Later, we will introduce what will be called a family of expander graphs, which is an infinite set of graphs satisfying some conditions. The goal of this paper is to provide constructions of error correcting codes. We will supply two constructions, both of which rely on families of expander graphs.

Chapter 2

Eigenvalues of a Graph

There is a virtuous way to link graphs to linear algebra, more specifically, we can relate a graph to a matrix called its adjacency matrix. This is indeed a very useful approach, since by studying the matrix and its eigenvalues we can infer properties about the graph. In this chapter we introduce all definitions regarding this connection, and we give a few results.

2.1 Adjacency Matrix of a Graph

We begin by giving the formal definition of the forementioned adjacency matrix.

Definition 2.1.

*Let G be a graph with n vertices, labeled $V = \{u_1, \ldots, u_n\}$ from 1 to n. The **adjacency matrix** of G, denoted $A(G)$, is a $n \times n$ matrix whose (i, j) entry is the number of edges in G between u_i and u_j.*

We may refer to the entry (i, j) as the entry (u, v), where u and v are the i-th and j-th vertices in the labeling. Also, when the context is clear, we denote the matrix as $A = A(G)$.

This matrix has real entries and is clearly symmetric, hence A has n real eigenvalues, which we denote by $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_n$, and we can also associate to it an orthonormal system of eigenvectors $e_1, \ldots, e_n$ with $Ae_i = \lambda e_i$. We refer to the eigenvalues of $A(G)$ as the **spectrum** of the graph G. As mentioned before, we can deduce properties of the graph from the eigenvalues of the matrix, here are some examples for a d-regular graph.

Remark 2.2.

- *$\lambda_1 = d$, and its corresponding eigenvector is $e_1 = (1/\sqrt{n}, \ldots, 1/\sqrt{n})$.*
- *The graph is connected if and only if $\lambda_1 > \lambda_2$.*
- *The graph is bipartite if and only if $\lambda_1 = -\lambda_n$.*

We introduce as well a more general definition of adjacency matrix.

Definition 2.3.

*Let $G = (V, E)$ be a graph, $L, R \subset V$ with $|L| = l$ and $|R| = r$. The **sets-adjacency matrix between L and R**, denoted $A(L, R)$, is a $l \times r$ matrix whose (u, v) entry is the number of edges in G between u and v (where $u \in L$ and $v \in R$).*

Notice that the adjacency matrix is a particular case of the set-adjacency matrix, since $A(G) = A(V(G), V(G))$. This second definition will come in handy in some cases, especially when working with bipartite graphs.

Given a d-regular graph G with n vertices, we denote $\lambda = \lambda(G) = \max\{|\lambda_2|, |\lambda_n|\}$. Since $\lambda_2 \geqslant \ldots \geqslant \lambda_n$, we have that λ is the largest absolute eigenvalue apart from λ_1. We are now in position to prove the following useful bound.

Lemma 2.4. *(Expander Mixing Lemma)*

Let G be a d-regular graph with n vertices, then for all $S, T \subset V$ we have

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

Proof.

Let 1_S and 1_T be the characteristic column vectors of S and T, and expand them in the orthonormal basis of eigenvectors, $1_S = \sum_i \alpha_i v_i$ and $1_T = \sum_j \beta_j v_j$. We obtain

$$|E(S,T)| = 1_S^{\mathrm{t}}\ A\ 1_T = (\sum_{i=1}^{n} \alpha_i v_i)\ A\ (\sum_{j=1}^{n} \beta_j v_j).$$

This gives $\sum_i \lambda_i\ \alpha_i\beta_i$ from the orthogonality of the vectors. Using Remark 2.2 we have that

$$\alpha_1 =< 1_S, e_1 >= \frac{|S|}{\sqrt{n}},\ \beta_1 = \frac{|T|}{\sqrt{n}},\ \text{and } \lambda_1 = d,$$

where $e_1 = (1/\sqrt{n}, \ldots, 1/\sqrt{n})$. Therefore

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

which by definition of λ gives

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

Using Cauchy-Schwartz we obtain the result

$$\left| |E(S,T)| \;-\; d\frac{|S||T|}{n} \right| \;\leqslant\; \lambda\ \|\alpha\|\,\|\beta\| \;\leqslant\; \lambda\ \sqrt{|S||T|}.$$

□

Notice that the term $d\frac{|S||T|}{n}$ is the expected number of edges between S and T in a d-regular graph. The lemma tells us that if λ is small, then so is the difference between the actual number of edges in the graph G and the expected number of edges if it was a random graph. The graph is almost random in this sense. The converse of this Lemma is also true, we state it without proof.

Lemma 2.5. *(Converse of the Expander Mixing Lemma)*

Let G be a d-regular graph with n vertices. Suppose that for every two disjoint sets $S, T \subset V$ we have

$$\left| |E(S,T)| \; - \; d\frac{|S||T|}{n} \right| \; \leqslant \; \rho\sqrt{|S||T|},$$

for some real positive ρ, then $\lambda \leqslant O(\rho + \rho\log(\frac{d}{p}))$.

Proof.

A proof of the previous Lemma can be found in [2]. □

2.2 Matrices Associated to a Graph

The classical Laplace operator is defined as $\Delta(f) = \operatorname{div}(\operatorname{grad}(f))$, we will define an analogue linked to graphs, the discrete Laplacian, which we will find useful. First we must define the analogues to the gradient and the divergence, then the form of the Laplacian will turn up naturally.

Let $G = (V, E)$ be a directed graph, then using the orientation of its edges we define K to be the $V \times E$ incidence matrix of G, whose entries are

$$K_{u,e} = \begin{cases} +1 & \text{if the edge } e \text{ starts at the vertex } u, \\ -1 & \text{if the edge } e \text{ ends at the vertex } u, \\ 0 & \text{otherwise.} \end{cases}$$

Definition 2.6.

Let $f : V \to \mathbb{R}$ be a function on the vertices of G, which we view as a row vector indexed by V. The ***gradient operator*** *maps f to fK, a vector indexed by E.*

Note that it measures the change of f along the edges of the graph, if e is the directed edge (u, v), then

$$(fK)_e = f(u) - f(v).$$

One could think of it as a potential difference on the vertices.

Definition 2.7.

Let $g : E \to \mathbb{R}$ be a function on the edges of G, which we view as a column vector indexed by E. The ***divergence operator*** *maps g to Kg, a vector indexed by V.*

Considering g as a flow, the evaluation of the divergence at a vertex gives the total outbound flow

$$(Kg)_v = \sum_{e \text{ starts at } v} g_e - \sum_{e \text{ ends at } v} g_e.$$

We want to define the Laplacian for undirected graphs, however, the Laplacian is defined in terms of matrices that require an oriented graph to be well defined. Hence, given an undirected graph G we consider the directed graph G_2, such that it has the same vertex set as G_1 and its edges are in bijection with those of G, except that they have some arbitrary orientation assigned. The gradient and divergence matrices are defined on the auxiliary graph G_2, while the Laplacian is defined on the original graph G. Beforehand, we assert that the Laplacian is independent of the arbitrary orientation of the edges in G_2.

Definition 2.8.

Let $f : V \to \mathbb{R}$ be a function, the ***Laplacian of an undirected graph*** *G is the operator that maps f to $KK^T\ f$. The matrix $L = L_G = KK^T$ is called the Laplacian of the graph.*

One can easily observe that L is a $|V| \times |V|$ symmetric matrix with

$$L_{u,v} = \begin{cases} -1 & \text{if } uv \in E, \\ \deg(v) & \text{if } u = v. \end{cases}$$

Also, from the remark after Definition 2.6 of the gradient, we obtain

$$fLf^T = fKK^Tf^T = \|fK\|^2 = \sum_{uv \in E} (f(u) - f(v))^2. \tag{2.1}$$

We can observe from this expression that the discrete Laplacian does not depend on the direction of the edges used to define the gradient and the divergence.

2.3 Ramanujan Graphs

Now that we have introduced the concept of adjacency matrix and the eigenvalues of a graph, we can now define a new family of graphs. A Ramanujan graph, named after Srinivasa Ramanujan, is a regular graph whose difference between the first and second eigenvalues of its adjacency matrix is almost as large as possible.

Definition 2.9.

Let G be a connected d-regular graph with n vertices, and let $\lambda_1 \geqslant \ldots \geqslant \lambda_n$ be the eigenvalues of its adjacency matrix $A(G)$. Whenever there exists λ_i with $|\lambda_i| < d$, define

$$\lambda(G) = \max_{|\lambda_i|<d} |\lambda_i|.$$

A ***Ramanujan graph*** *is a d-regular graph for which $\lambda(G)$ is defined and*

$$\lambda(G) \leqslant 2\sqrt{d-1}.$$

We will mention a few easy examples of graphs that satisfy this condition, and therefore are Ramanujan graphs.

Example 2.10.

- *The complete graph K_3 has the adjacency matrix*

$$A(K_3) = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix},$$

and hence its characteristic polynomial is $\lambda^3 - 3\lambda - 2$. Then, the eigenvalues of the graph K_3 are -1 and 2, which means that $\lambda(K_3) = 1$. Since K_3 is a 2-regular graph, so $d = 2$, and we clearly obtain that it satisfies $\lambda(K_3) \leqslant 2\sqrt{d-1}$.

- *In general, the complete graph K_n is an $(n-1)$-regular graph, whose characteristic polynomial is*

$$(\lambda - d)(\lambda + 1)^d,$$

where $d = n - 1$. Thus $\lambda(K_n) = 1$, and we can see that the complete graph satisfies $\lambda(K_n) \leqslant 2\sqrt{d-1}$ as long as $d \geqslant 2$, i.e. for any $n \geqslant 3$.

- *Another example of a Ramanujan graph is the complete bipartite graph $K_{r,r}$. It is a r-regular graph whose adjacency matrix has only eigenvalues are $r, -r$ and 0. It can be verified that its characteristic polynomial is*

  $$(\lambda - r)(\lambda + r)\lambda^{2r-2}.$$

  Hence, $\lambda(K_{r,r}) = 0$ for any $r \geqslant 2$ (the graph itself has at least 4 vertices). From this it is obvious that it satisfies the condition to be a Ramanujan graph.

In the next chapter we will introduce the concept of expansion of a graph, which briefly explained measures how well connected a graph is. It will then become apparent that Ramanujan graphs have good expansion, it is therefore not strange that our examples of Ramanujan graphs were the complete graphs, and its bipartite analogous.

As we will later see, it is important to us not only to define families and classes of graphs, but to provide constructions of elements of the families. It will be particularly important to have constructions of elements of a family of graphs of constant degree d and arbitrary size n. Although we have already mention an infinity of Ramanujan graphs, they are all of strictly increasing degree. Since Ramanujan graphs will be used in one of the constructions of error correcting codes, we briefly outline one of the explicit constructions of Ramanujan graphs of some constant degree. The construction is due to Lubotzky, Phillips and Sarnak [5].

Let p and q be distinct primes with $p \equiv q \equiv 1 \mod 4$, and let u be an integer such that $u^2 \equiv -1 \mod q$. The equation $a^2 + b^2 + c^2 + d^2 = p$ is known to have $8(p+1)$ solutions, among which there are exactly $p+1$ with $a > 0$ and b, c, d even. To each such solution $v = (a, b, c, d)$ we associate the matrix

$$V_v = \begin{pmatrix} a + ub & c + ud \\ -c + ud & a - ub \end{pmatrix},$$

which gives us a total of $p+1$ matrices in $PGL_2(\mathbb{Z}/q\mathbb{Z})$. From this matrices it is possible to construct a Ramanujan graph G, which is a $(p+1)$-regular graph, as seen in [5]. Then, given that there are an infinite number of primes q congruent to 1 modulo 4, we get an infinite family of Ramanujan graphs of constant degree $p+1$.

Chapter 3

Expansion of a Graph

For a given subset of vertices of a graph G, we could wonder about of how well it connects to the rest of the graph, that is the concept of expansion, and there are many different ways to measure it. Although there are other, we will mainly focus on the approach of expansion by edges. Then a family of graphs such that it satisfies some conditions on our definition of expansion will be called a family of expander graphs.

In this chapter we start with some formal definitions, then using the associated matrices of a graph we will derive some important results on expansion, which will show that the eigenvalues are closely related to expansion, particularly the first two eigenvalues and its difference which is known as spectral gap.

3.1 Expander Graphs

Definition 3.1.

*The **edge expansion rate** of a graph $G = (V, E)$ is defined as*

$$h(G) = \min_{\{S \subset V : |S| \leqslant \frac{|V|}{2}\}} \frac{|\partial S|}{|S|}.$$

We may denoted it simply as h if the context is clear. We now introduce the concept of expander graphs.

Definition 3.2.

*A sequence of d-regular graphs $\{G_i\}_{i \in \mathbb{N}}$ of increasing size with i is a **family of expander graphs** if there exists a $\varepsilon > 0$ such that $h(G_i) \geqslant \varepsilon$ for all i.*

3.2 Bounds on the Expansion Rate

In this chapter we present a lower and upper bound on the edge expansion rate $h(G)$ of a graph d-regular graph G, this bounds relate the expansion of the graph to its eigenvalues. The bounds that we will prove are

$$\frac{d - \lambda_2}{2} \leqslant h(G) \leqslant \sqrt{2d\,(d - \lambda_2)}, \tag{3.1}$$

where the spectrum of G is $\lambda_1, \ldots, \lambda_n$. Since for d-regular graphs $\lambda_1 = d$, we observe that the inequality depends only on the first two eigenvalues of the graph. The quantity $d - \lambda_2$ is known as the **spectral gap**.

In order to prove this result we fist prove some lemmas.

Lemma 3.3.

Let $G = (V, E)$ be a simple connected (n, d)-graph. Then we have

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

where $h(G)$ is the expansion of the graph, and λ_2 is its second largest eigenvalue.

Proof.

Let $S \subset V$ be a subset such that $h(G) = |\partial S|/|S|$ with $|S| \leqslant n/2$. Consider the vector $f = |\overline{S}|\ 1_S - |S|\ 1_{\overline{S}}$, where 1_X denotes the characteristic vector of the set X. We compute

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

Let $A = (a_{i,j})$ be the adjacency matrix of G, hence

$$fAf^T = \sum_{i,j=1}^{n} f_i a_{i,j} f_j = \sum_{ij \in E} f_i f_j = 2(|E(S)||\overline{S}|^2 + |E(\overline{S})||S|^2 - |S||\overline{S}||\partial S|).$$

Using that G is d-regular we further obtain the equalities

$$2|E(S)| = d|S| - |\partial S|,$$
$$2|E(\overline{S})| = d|\overline{S}| - |\partial S|.$$

Finally we clarify that

$$fAf^T = f_2^2\lambda_2 + \ldots + f_n^2\lambda_n \leqslant \lambda_2 \|f\|^2.$$

Using all the previous calculations, together with the fact that $h(G) = |\partial S|/|S|$ and $|\overline{S}| \geqslant n/2$ provides

$$\lambda_2 \geqslant \frac{fAf^T}{\|f\|^2} = \frac{nd|S||\overline{S}| - n^2|\partial S|}{n|S||\overline{S}|} = d - \frac{n|\partial S|}{|S||\overline{S}|} \geqslant d - 2h(G),$$

and solving the inequality for $h(G)$ proves the lemma. □

The previous is used to prove the lower bound, while the proof of the upper bound will be divided in two lemmas, but before we continue we present some notation. We denote the set $\{1, 2, \ldots, n\}$ as $[n]$. For a vector $v = (v_1, \ldots, v_n)$, its **positive part** v^+ consists of replacing its negative entries with zeroes and leaving the rest as it is, or more clearly $v_j^+ = \max\{0, v_j\}$. The **support** of the vector v, denoted $\operatorname{supp}(v)$, is the set of indexes where v_j is not zero, explicitly $\operatorname{supp}(v) = \{j \in [n] : v_j \neq 0\}$. We now proceed to prove the two following lemmas.

Lemma 3.4.

Let $G = (V, E)$ be a simple connected (n, d)-graph. Then there exists a function f such that

$$\frac{fLf^T}{\|f\|} \leqslant d - \lambda_2,$$

where L is the Laplacian of the graph, and λ_2 is its second largest eigenvalue.

Proof.

Let g be an eigenvector associated to λ_2, and let $f = g^+$ and $V^+ = \operatorname{supp}(f)$. We can assume that V^+ has at most $n/2$ vertices (otherwise take $-g$ instead of g and redefine f and V^+ accordingly).

Observe that for $x \in V^+$ we have

$$\begin{aligned}(Lf)_x &= df_x - \sum_{y \in V} a_{xy} f_y = dg_x - \sum_{y \in V^+} a_{xy} g_y \\ &\leqslant dg_x - \sum_{y \in V} a_{xy} g_y = (Lg)x = (d - \lambda_2) g_x,\end{aligned}$$

thus $(Lf)_x \leqslant (d-\lambda_2)g_x$. Now we calculate

$$\begin{aligned} fLf^T &= \sum_{x\in V} f_x(Lf)x \;\leqslant\; \sum_{x\in V} f_x(d-\lambda_2)g_x \;=\; \sum_{x\in V^+} g_x^2(d-\lambda_2) \\ &= (d-\lambda_2)\sum_{x\in V^+} g_x^2 \;=\; (d-\lambda_2)\sum_{x\in V} f_x^2 \;=\; (d-\lambda_2)\left\|f\right\|^2, \end{aligned}$$

and therefore $fLf^T \leqslant (d-\lambda_2)\left\|f\right\|^2$. Solving the inequality for the spectral gap gives the desired result. □

Lemma 3.5.

Let $G=(V,E)$ be a simple connected (n,d)-graph, and let f be the function used in the proof of the previous lemma. Then f satisfies

$$\frac{h(G)^2}{2d} \leqslant \frac{fLf^T}{\left\|f\right\|^2},$$

where L is the Laplacian of the graph, $h(G)$ is its edge expansion rate, and λ_2 is its second largest eigenvalue.

Proof.

We define the auxiliary quantity B_f, which depends only on f, as

$$B_f = \sum_{xy\in E} |f_x^2 - f_y^2|.$$

We will estimate its value in two different ways to arrive to the desired inequality.

First, using the Cauchy-Schwartz inequality we get

$$B_f = \sum_{xy\in E} |f_x+f_y||f_x-f_y| \leqslant \sqrt{\sum_{xy\in E}(f_x+f_y)^2}\cdot\sqrt{\sum_{xy\in E}(f_x-f_y)^2}. \tag{3.2}$$

For the left square root we obtain

$$\begin{aligned}
\sqrt{\sum_{xy\in E}(f_x+f_y)^2} &= \sqrt{\sum_{xy\in E}(f_x^2+f_y^2+2f_xf_y)} \\
&\leqslant \sqrt{2\sum_{xy\in E}(f_x^2+f_y^2)} \qquad (3.3)\\
&= \sqrt{2d\sum_{x\in V}f_x^2} \quad = \quad \sqrt{2d}\,\|f\|\,.
\end{aligned}$$

While for the right square root, using (2.1), one obtains

$$\sqrt{\sum_{xy\in E}(f_x-f_y)^2} = \|fK\|\,. \qquad (3.4)$$

The equations (3.2), (3.3) and (3.4) give the upper bound $B_f \leqslant \sqrt{2d}\,\|f\|\,\|fK\|$.

Now, we proceed to estimate B_f in a different way. Before we begin, without loss of generality, consider the f_i's with the ordering $f_{i+1} \leqslant f_i$ for all $1 \leqslant i \leqslant n-1$. The second estimate is

$$\begin{aligned}
B_f \quad &= \sum_{\substack{xy\in E,\\ x<y}}(f_x^2-f_y^2) \;=\; \sum_{\substack{xy\in E,\\ x<y}}\left(\sum_{i=x}^{y-1}(f_i^2-f_{i+1}^2)\right) \\
&= \sum_{i=1}^{n-1}(f_i^2-f_{i+1}^2)\;|E([i],\overline{[i]})| \;\geqslant\; \sum_{i\in V^+} i\;h(G)(f_i^2-f_{i+1}^2) \qquad (3.5)\\
&= h(G)\sum_{i\in V^+}f_i^2 \;=\; h(G)\,\|f\|^2\,.
\end{aligned}$$

We make an attempt to make more clear the step in line (3.5). We can change the indexing to V^+ because if $i \notin V^+$ then $f_i = 0$, and from the assumption $f_{i+1} \leqslant f_i$ we get that $f_{i+1} = 0$ as well. Since $|V^+| \leqslant n/2$, from the definition of the expansion rate we obtain $h(G) \leqslant |E([i],\overline{[i]})|/|[i]|$, and using that $|[i]| = i$ gives the the bound $i\;h(G)$ at the mentioned step.

This implies that $h(G)\left\|f\right\|^2 \leqslant B_f$. This in conjunction with the forementioned upper bound gives $h(G)\left\|f\right\|^2 \leqslant B_f \leqslant \sqrt{2d}\left\|f\right\| fLf^T$, which in turn gives our desired result. □

We restate the equation (3.1) as a theorem bounding the edge expansion rate with the spectral gap.

Theorem 3.6.

Let $G = (V, E)$ be a simple connected d-regular graph, its edge expansion satisfies

$$\frac{d-\lambda_2}{2} \leqslant h(G) \leqslant \sqrt{2d\,(d-\lambda_2)}.$$

Proof.

Lemma 3.3 gives the lower bound, while together Lemma 3.4 and Lemma 3.5 give the upper bound. □

Chapter 4

Zig-Zag Products

This chapter will introduce the different products between graphs, i.e. the different ways in which we can obtain a new graph from previously known ones. The graphs constructions in this chapter will be later used to construct families of expander graphs.

4.1 Replacement Product

The replacement product, denoted by $\textcircled{r}$, is an asymmetric binary operation between graphs. It is the product of an (n, m)-graph G and an (m, d)-graph H, whose result is a graph with nm vertices and $(d + 1)nmd/2)$ edges.

Intuitively is the original graph G, except that each vertex is replaced by a copy of the graph H. We give the formal definition.

Definition 4.1.

The graph $G \textcircled{r} H$, called the ***replacement product*** *between the graphs G and*

H, has as vertex set $V(G) \times V(H)$, and $((u,i),(v,j))$ is in the edge set if $(u,v) \in G$ and $(i = j)$, or if $(i,j) \in H$.

4.2 General Zig-Zag Product

The zig-zag product, denoted by $\textcircled{z}$, is an asymmetric binary operation between graphs. It is the product of an (n,m)-graph G and an (m,d)-graph H, whose result is an (nm, d^2)-graph.

For an intuitive idea one should think of the replacement product between the graphs G and H, where there will be an edge between the vertices x and y if there is a path between them that takes an edge in one of the copies of H, followed by an edge originated from the graph G, and finally one last edge on one of the copies of H. Hence, perhaps it is better to think of it as a "zig-zag-zig" product, as the edges of the product correspond to paths of edges on the graphs $H - G - H$. We give the formal definition.

Definition 4.2.

*The graph $G \textcircled{z} H$, called the **zig-zag product** between the graphs G and H, has as vertex set $V(G) \times V(H)$, and $((u,i),(v,j))$ is in the edge set if there are k, l such that $(i,k),(l,j) \in H$ and $e_v^k = e_u^l$.*

4.3 Zig-Zag Product for Bipartite Graphs

Finally we will present a small variation of the zig-zag product, this one is specially defined for bipartite graphs. This definition is different from the previous one because the graph does not need to be in general d-regular, as we ask for the condition of regularity to be satisfied only on one of the sides of the graph. This could in turn cause, in the standard definition of the zig-zag product, some of the vertices to not able to do a zig-zag-zig and have neighbors, i.e. there could be isolated vertices in the resulting product. As far as a definition goes this would not cause any problems, but having an isolated vertex automatically causes a graph to have an edge expansion rate of 0, something highly undesirable for our constructions. Intuitively the zig-zag product for bipartite graph is the same as for general graphs, except that we erase the isolated vertices from the final product.

Definition 4.3.

*For H a d-regular bipartite graph with s vertices on each side, and G an s-regular bipartite graph with n vertices on each side, the **bipartite zig-zag product** $G \circledz H$ has vertex set $V(G) \times V(H)/2$, and $((u,i),(v,j))$ is in the edge set if $(u,v) \in V(G)$, $(i,j) \in H$ and $i < j$.*

An alternative definition is to consider the graph H to be a directed graph.

Chapter 5

The Margulis Expander Graphs

In this chapter we will introduce a new way to construct graphs from algebraic structures. The first explicit constructions of expander graphs were found from this approach, namely, the Margulis construction. The Margulis construction will only define a family of graphs, we will then proceed to sketch a proof that in this family is in fact a family of expander graphs.

5.1 Margulis Construction

We briefly describe one graph which is an infinite analog of the Margulis construction.

Example 5.1.

- *Let $G = (V, E)$ be the infinite graph with vertex set $V = I \times I$, where $I = [0, 1)$ is the half-open unitary interval. Define the linear transformations $T, S : V \to E$ as*

$$T(x,y) = (x+y,y) \mod 1, \textit{ and } S(x,y) = (x,x+y) \mod 1.$$

The neighbors of a vertex $(x,y) \in V$ *are* $T(x,y), T^{-1}(x,y), S(x,y), S^{-1}(x,y)$.

Note that this is an undirected 4-regular graph. Also note that the graph includes multiple edges and loops (even multiple loops). Consider for example the neighbors of the vertex $(0,0)$, which are the same vertex four times. It is a graph in which the edges are defined by some transformations and its inverses.

Considering the finite-graph case of this example yields the construction given by Margulis. Further considering infinite different sets of vertices in the construction gives an infinite amount of graphs, thus a family of graphs. We will later prove that this graphs form a family of expander graphs.

Definition 5.2.

Let G_n *be a family of 8-regular graphs. The vertex set of* G_n *is* $V_n = \mathbb{Z}_n \times Z_n$. *Define the matrices and vectors*

$$T_1 = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}, \; T_2 = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}, \; e_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \; \textit{and} \;\; e_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

The neighbors of a vertex $v = (x,y) \in V$ *are* $T_1v, T_1v + e_1, T_2v, T_2v + e_2$*, and the corresponding four inverse transformations. All operations are done in* $\mathbb{Z}_n$*, i.e. the calculations are* $\mod n$*. This are known as the* ***Margulis graphs****.*

Note that this is an undirected 8-regular graph, and resembling the previous example, it may have multiple edges or loops.

5.2 Margulis as a Family of Expander Graphs

In this section we will prove that the family of Margulis graphs forms a family of expander graphs. This amounts to prove that there exists an $\varepsilon > 0$ such that for all graphs G_n in the family we have $h(G_n) > \varepsilon$.

The original proof of expansion by Margulis was based on representation theory, it was existential and did not provide any specific bound on the edge expansion rate h. A specific bound was derived by Gabber and Galil using harmonic analysis, and later improved by Jimbo and Marouka using Fourier analysis. We give here a simplified proof due to Boppana.

Gabber and Gail proved that the graphs G_n satisfy $\lambda_2 \leqslant 5\sqrt{2} < 8$ for every positive integer n. Using Theorem 3.6 it can be then deduced that the defined family forms a family expander graphs. We will prove a slightly weaker result, that $\lambda_2 \leqslant 3.65$, but which still suffice for our purposes. The proof is long and ingenious, but before providing it we state without proof the following Lemma.

Lemma 5.3.

There is a partial order on $\mathbb{Z}_n^2$ such that for every z inside the diamond, either

Three of the four points $T_1z, T_2z, T_1^{-1}z$ and $T_2^{-1}z$ are $> z$ and one is $< z$.

Two of the four points $T_1z, T_2z, T_1^{-1}z$ and $T_2^{-1}z$ are $> z$ and the other two are incomparable with z.

Where the diamond is the region comprehended inside the rhombus with corners $(n/2, 0), (0, n/2), (-n/2, 0)$, and $(0, -n/2)$.

Theorem 5.4.

The graph G_n, as defined by Margulis, satisfies $\lambda_2(G_n) \leqslant \delta < 8$ for some constant $\delta > 0$ and for every positive integer n.

Proof.

We will actually prove this result for $\delta = 3.65$ which is greater than $5\sqrt{2}$, thus a weaker result.

By the formulas on eigenvalues used in Theorem 3.6, we can change the statement to functions, such that if $f : \mathbb{Z}_n^2 \to \mathbb{R}$ satisfies $\sum_{x \in Z_n^2 f(x)=0}$ then

$$\sum_{xy \in E} f(x)f(y) \leqslant \delta \sum_{x \in V} f(x)^2.$$

Recalling that the neighbors of a vertex $x \in V$ are $T_1 x, T_1 x + e_1, T_2 x, T_2 x + e_2$ and its inverses, and that for such inverses there must be a $y \in V$ such that the transformation gives x as a result. Then we can restate the last inequality as

$$\sum_{z \in V} f(z) \cdot [f(T_1 z) + f(T_1 z + e_1) + f(T_2 z) + f(T_2 x + e_2)] \leqslant \delta \sum_{z \in V} f^2(z).$$

Let F denote the Fourier transform of f, and $w = e^{\frac{2\pi i}{n}}$ a n-th primitive root of unity. Now, we take the Fourier transform, we use Parseval's identity and the shift property, i.e. if $g(x) = f(Ax+b)$ then $\hat{g}(x) = w^{-<A^{-1}b,x>} \hat{f}((A^{-1})^T x)$. Then our claim is satisfied if for all $F : Z_n^2 \to \mathbb{C}$ with $F(0,0) = 0$ we have

$$\left| \sum_{z=(z_1,z_2) \in \mathbb{Z}_n^2} \overline{F(z)} \cdot [F(T_2^{-1} z)(1 + w^{-z_1}) + F(T_1^{-1} z)(1 + w^{-z_2})] \right| \leqslant \delta \sum_{z \in Z_n^2} |F(z)|^2.$$

We define $G : \mathbb{Z}_n^2 \to \mathbb{R}$ as $G = |F|$. Using that the new function is non-negative, the triangle inequality, and the identity $|1 + w^{-t}| = 2\ |cos(\frac{\pi t}{n})|$, we further restate our claim to non-negative functions G with $G(0,0) = 0$ satisfying

$$\sum_{z=(z_1,z_2) \in \mathbb{Z}_n^2} 2G(x) \cdot [G(T_2^{-1} z)|cos(\frac{\pi z_1}{n})| + G(T_1^{-1} z)|cos(\frac{\pi z_2}{n})|] \leqslant \delta \sum_{z \in Z_n^2} |G(z)|^2. \quad (5.1)$$

In what follows, we will bound $2G(z)$ by using the arithmetic inequality

$$2ab \leqslant c\ a^2 + c^{-1}\ b^2, \tag{5.2}$$

which holds for any real numbers $a, b, c \in \mathbb{R}$ with $a, b \geqslant 0$ and $c > 0$. However, it is not possible to bound $2G(z)$ with a fixed value of c in the inequality, for example $\gamma = 1$ gives a bound with a coefficient of 8 and we require $\delta < 8$. Instead we will consider a variable $\gamma : \mathbb{R}^2 \to \mathbb{R}$ such that $\gamma\gamma^{-1} = 1$ in all its domain. Taking the same partial ordering used in Lemma 5.3 we define γ to be

$$\gamma = \begin{cases} \alpha & \text{if } (x_1, y_1) > (x_2, y_2), \\ \frac{1}{\alpha} & \text{if } (x_1, y_1) < (x_2, y_2), \\ 1 & \text{otherwise.} \end{cases}$$

It is here that for simplicity we define $\alpha = 5/4$, and it is this that determines the bound with $\delta = 3.65$. A different choice of α and a more in depth analysis than what we are about to do would give the bound with $\delta = 5\sqrt{2}$.

One can easily deduce that the way γ has been defined implies $\gamma(x, y) \cdot \gamma(y, x) = 1$, for all $x, y \in \mathbb{Z}_n^2$. From (5.2) we obtain

$$2 \cdot G(x)G(y) \leqslant \gamma(x, y) \cdot G^2(x) + \gamma(y, x) \cdot G^2(y),$$

and applying this in (5.1) one obtains the upper bound

$$\sum_{z=(z_1,z_2)\in Z_n^2} |cos(\frac{\pi z_1}{n})|\cdot[\gamma(z,T_2^{-1}z)G^2(z)+\gamma(T_2^{-1}z)G^2(T_2^{-1}z)]+|cos(\frac{\pi z_2}{n})|\cdot[\gamma(z,T_1^{-1}z)G^2(z)+\gamma(T_1^{-1}z)G^2(T$$

Then from the definitions of the transformations we see that z_1 is invariant under T_2 and z_2 under T_1, hence the last equation transforms into

$$\sum_{z=(z_1,z_2)\in Z_n^2} G^2(z)\left[|cos(\frac{\pi z_1}{n})| \cdot [\gamma(z, T_2^{-1}z) + \gamma(T_2^{-1}z)] + |cos(\frac{\pi z_2}{n})| \cdot [\gamma(z, T_1^{-1}z) + \gamma(T_1^{-1}z)]\right].$$

For a given z, proving that the corresponding term in the sum is bounded by $\alpha G^2(z)$ would yield the desired result. Thus, now it suffices to prove

$$|cos(\frac{\pi z_1}{n})| \cdot [\gamma(z, T_2^{-1}z) + \gamma(T_2^{-1}z)] + |cos(\frac{\pi z_2}{n})| \cdot [\gamma(z, T_1^{-1}z) + \gamma(T_1^{-1}z)] \leqslant \frac{\alpha}{2}. \quad (5.3)$$

We consider the two possible cases.

Outside the diamond. Suppose without loss of generality that we are in the first quadrant. Since $cos(\frac{\pi z_2}{n})$ is decreasing, then $|cos(\frac{\pi z_1}{n})+cos(\frac{\pi z_2}{n})|$ has its maximum on the boundary of the diamond, where $cos(\frac{\pi z_2}{n}) = sin(\frac{\pi z_1}{n})$. Hence, $cos(\frac{\pi z_1}{n})+cos(\frac{\pi z_2}{n}) = cos(\frac{\pi z_1}{n}) + sin(\frac{\pi z_1}{n}) \leqslant \sqrt{2}$. Overestimating γ by 5/4, we obtain the bound needed in (5.3) outside the diamond.

Inside the diamond. We bound the cosines by 1, and thus it suffices to prove the inequality

$$\gamma(z, T_1 z) + \gamma(z, T_1^{-1}z) + \gamma(z, T_2 z) + \gamma(z, T_2^{-1}z) + \leqslant \delta.$$

Then by Lemma 5.3, if two points are $> z$ and the rest are incomparable, then our equation is bounded by 3.6. On the other hand, if three points are $> z$ and one is $< z$, then it is bounded by 3.65. Therefore, we have obtain again the bound needed in 5.3 inside the diamond. This concludes the proof of the theorem. □

Chapter 6

Coding Theory

In order to effectively exchange information between two individuals, the sender and the receiver of the messages, one would like to have very clever and efficient methods for them to communicate. However, due to external factors independent to the method, there could be differences between the sent message and the one received. Then, it is further desirable that the method could also detect and correct these errors, at least up to some extent.

The analysis of transmitting information effectively and efficiently is the field of study of coding theory, and algorithms able to accomplish what we described are called **error correcting codes**. These are very important as they are widely used in our daily lives, to mention only a few examples, they are used in TV transmissions, in cell phone calls, and between connected computers over the internet. There are error correcting codes being implemented even between the user himself and the computer.

In this chapter we introduce the basics of coding theory, giving the definition of codes in general, their distance and rate, and the special class of linear codes. In the next chapter we will utilize this concepts and proceed to build codes from graphs.

6.1 Introduction to Coding

Consider the problem of sending information from individual A to individual B through some medium. The information that we want to transmit is called the **message**, while the individuals are the **sender** and **receiver** respectively, and the medium is called **noisy channel**.

Instead of trying to send the original message, which could be of any nature, it is convenient to send numeric bits of information in its representation, which is known as an **encoding** of the message. Upon reception the inverse is done to retrieve the original message, known as **decoding**. We will restrict ourselves to schemes that transmit the information in n-bit blocks, and we only consider the binary case.

We consider that the noisy channel, through which the data transfer is done, could cause some error changes in the bits transmitted. To solve this problem and make sure the receiver can correct the errors, we agree in advance that only a subset, sometimes called **dictionary**, of the 2^n possible n-bit blocks would be transmitted.

We would then define some sense of distance between the elements of the dictionary. This way upon the reception of an encoded word, if it is in the dictionary we accept it, otherwise we are detecting there was an error in the transmission, and the natural error correcting method would be to find the word in the dictionary which is closest to the one received. This is the approach which was suggested by Shannon. We formulate this last concepts formally.

Definition 6.1.
A ***code*** *C is a set of n-bit binary strings, and its elements are called* ***codewords****.*

The metric used in the code is the Hamming distance.

Definition 6.2.

*The **Hamming distance** between $x, y \in \{0,1\}^n$, denoted d_H, is the number of coordinates on which x and y differ,*

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

We will denote it simply as $d(x, y)$ if the context is clear.

We want two conflicting properties in a code. On one hand, we want codewords to be distant from each other, this is to ensure that we will be able to perform a decent error correcting routine. On the other hand, we want many codewords, this is to increase the efficiency of the code by having more possible transmittable encoded messages in each n-bit string sent.

In other words, we want distant codewords, but not so distant that the maximum code size is small. One might intuitively observe how this is related to the conflicting properties of expansion in a graph. This notions are enclosed in the following definitions.

Definition 6.3.

*The **distance of a code** $C \subset \{0,1\}^n$, denoted dist(C), is the minimum Hamming distance between a pair of distinct codewords $x, y \in C$, formally*

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

*The **rate** of a code C is defined as*

$$\mathrm{rate}(C) = \frac{\log |C|}{n}.$$

We might sometimes consider instead the **relative distance** of the code, which is $\delta(C) = \mathrm{dist}(C)/n$.

As mentioned before, our aim is to have codes with both maximized distance and rate. Our problem has been reduced to finding a good code with the two conflicting properties. Observe that in practice a code should have efficient coding, decoding, and error correcting routines. A code with efficient routines is consider to be a good code, even further a family of codes is asymptotically good if it satisfies the following basic requirements.

Definition 6.4.

A family of codes $C_n \subset \{0,1\}^n$ *is* ***asymptotically good*** *if there exists some fixed constants* $d > 0$ *and* $r > 0$*, such that for all* n *we have both* $\delta(C) > d$ *and* $\mathrm{rate}(C) > r$*. The family is called* ***efficient*** *if the encoding and decoding routines can be performed in polynomial time in* n*.*

The existence of asymptotically good codes can be proved with a probabilistic argument, however, the search for good efficient codes took over two decades. We will provide explicit constructions of asymptotically good codes in the next chapter. First we need to introduce a useful type of codes, those that form linear vector subspaces of $\{0,1\}^n$.

Definition 6.5.

A code $C \subset \{0,1\}^n$ *is called a* ***linear code*** *of dimension* k *and length* n*, if it is a* k*-dimensional vector subspace of* $\{0,1\}^n$*.*

Note that a linear code of dimension k has rate k/n. Such codes can be described concisely by specifying a basis, hence they can be encoded in $O(n^2)$ time. We will later see that with our construction of codes from graphs, we can achieve linear time $O(n)$ in the decoding. For now, we proceed to define the Hamming weight in the vector space.

Definition 6.6.

The ***Hamming weight*** *of* $x \in F_2^n$ *is defined as*

$$w(x) = d(x, 0).$$

Basically, the Hamming weight of a vector is the number of 1's in it. It is easy to deduce, by linearity, that the distance of a code equals the smallest weight of a non-zero codeword.

Since a linear code can be specified by giving a basis for the k-dimensional subspace $C \subset F_2^n$, we can then view C as the right kernel of

$$C = \{x \ : \ H_C x = 0\},$$

for some $(n-k) \times n$ matrix H_C, and where the operations are done in F_2. The matrix H_C is called a **parity check** matrix for C, and we may denote it simply as H if the context is clear.

6.2 Asymptotic Bounds

The two wanted conflicting properties, the distance and the rate of a code, depend strongly on each other. We devote this section to stating some of the well known bounds between them. First we introduce some definitions.

The **Hamming ball** with center $x \in F_2^n$ and radius r is the set

$$B(x, r) = \{y \in F_2^n \ : \ d(x, y) \leqslant r\}.$$

Note that its **volume**, the amount of points inside it, is equal to

$$v(n, r) = \sum_{i=0}^{r} \binom{n}{i}.$$

Now, we prove the **Gilbert-Varshamov bound**.

Theorem 6.7.

There exists a code of length n, distance at least d and size at least $2^n/v(n,d)$. The statement also holds in particular for linear codes.

Proof.

Consider the following greedy algorithm, that constructs a code C of length n and distance at least d.

Initialize with $S_1 = \{0,1\}^n$ and $C_1 = \emptyset$. Then, at the i-th step pick any point $x \in S_i$, then set $C_{i+1} = C_i \cup x$, and $S_{i+1} = S_i \setminus B(x,d)$. We continue this procedure until S_j is empty for some j.

Since the initial size of S_1 is 2^n, and at each iteration the size of the following S_i is reduced by at most $v(n,d)$, we obtain

$$j \cdot v(n,d) = |C| \cdot v(n,d) \geqslant 2^n,$$

which implies the wanted bound on the size. Clearly, the code $C = C_j$ has distance at least d by construction. This provides the result for general codes, we will now prove it for linear codes.

Since a linear code C of dimension k can be specified by a $(n-k) \times n$ parity check matrix H_C, and since its distance is the weight of the smallest non-zero codeword, then it is not hard to see that the distance of C is the smallest number of columns in H_C whose sum is the zero vector.

We will construct matrix one column at a time, such that at each step there is no dependent set of fewer than d columns. We can always add a j-th column satisfying this condition, granted that

$$\sum_{i=0}^{d-1} \binom{j-1}{i} < 2^{n-k},$$

due to the fact that the new column must be different from all possible sums of $d-1$ or less columns. The worst case scenario is when we are barely able to add the n-th column to build the matrix, hence obtaining

$$\sum_{i=0}^{d-1} \binom{n}{i} \geqslant 2^{n-k}.$$

The resulting code C has dimension at least k and so $|C| \geqslant 2^k$, which provides the bound on the size. The code has distance at least d by construction. □

We continue by proving the **sphere-packing** bound.

Theorem 6.8.
Any code C of length n and distance d satisfies $|C| \leqslant 2^n/v(n, d/2)$.

Proof.
For a code C of distance d, from the definition of its distance we observe that all balls of radius $d/2$ centered at codewords of C must be disjoint. The bound is then deduced by dividing the size of the space by the size of each such ball. □

Chapter 7

Codes from Graphs

The ultimate goal of this chapter is to explain how to create error correcting codes from graphs, for which we give two different constructions. The first construction uses bipartite graphs, and the second one uses regular graphs. The last construction was further improved by Gilles Zémorby using Ramanujan graphs, his construction is the one presented here. The codes arising from both these constructions are proved to be efficient error correcting codes.

7.1 First Construction

We are now in position to explain the connection between graphs and linear codes, then using expander graphs we will be able to construct good codes. This was one of the initial motivations for the definition of expanders and their explicit construction. The construction of codes that will be shown in this chapter is based in bipartite graphs.

Consider a bipartite graph $G = (L, R, E)$ and its adjacency matrix $A = A(L, R)$.

Suppose that $|L| = n$ and $|R| = m$, and that $n \geqslant m$, then A is a $m \times n$ matrix. We then use A as a parity check matrix to construct a linear code C of length n. The code constructed via the graph G is denoted as $C(G)$.

As we had explained in Chapter 1, the nature of the vertices of the graph is arbitrary. In this sense the left vertices of G, which belong to L, can be viewed as variables, while the right vertices belonging to R are the constraints, all of which gives rise to the linear vector subspace C.

We will introduce a variant of the concept of expansion for bipartite graphs, this is based in the vertex expansion.

Definition 7.1.

*Let $G = (L, R, E)$ be a bipartite graph, its **left vertex expansion ratio** is defined as*

$$L(G, d) = \min_{\{S \subset L \,:\, |S| \leqslant d\}} \frac{|\Gamma(S)|}{|S|}.$$

In other words, every subset of L of size at most d satisfies $|\Gamma(S)| \geqslant L(G, d)|S|$. Additionally, note that regardless of d $L(G, d)$ cannot exceed k for a k-left-regular graph.

As it can be seen in [4], the zig-zag product for bipartite graphs can be used to construct explicit k-left-regular bipartite graphs $G(L, R, E)$, with constant rate codes and left vertex expansion rate $L(G, d) > .9k$ for any $d \geqslant a \cdot n$ for some constant $a > 0$.

In short, we are saying that we can explicitly construct expander bipartite graphs, in the sense of left vertex expansion rate.

We will first show how these graphs generate linear codes which are asymptotically good codes, and afterwards we will demonstrate that they are also efficient. We will show that they can be decoded in polynomial time, and in fact linear time is achieved.

We do not worry much about encoding, since it can be done in $O(n^2)$ for being linear codes, and this is already in polynomial time. However, it can be proved that given the structure provided by the bipartite graphs, coding can also be done in linear time.

The following two theorems, due to Sipser and Spielman, prove our claims. First, that the codes from the bipartite graphs are asymptotically good codes, i.e. that the codes have large distance.

Theorem 7.2.
Let G be a k-left-regular bipartite graph, if $L(G,d) > k/2$, then $\operatorname{dist}(C(G)) \geqslant d$.

Proof.
We claim that for any subset $S \subset L$ of size $|S| \leqslant d$ there exists a unique neighbor to it in the right vertices, in other words, there is a vertex $v \in R$ such that $|\Gamma(x) \cap S| = 1$. To prove this consider that $|E(S, \Gamma(S))| = k|S|$ because the graph is k-left-regular, but at the same time $|\Gamma(S)| > k|S|/2$ because of the left vertex expansion rate $L(G,d) > k/2$.

Then for a vertex in $\Gamma(S)$ the average amount of edges from it to S is strictly less than 2, and since they all have at least 1 edge being that they are in the set of neighbors, then there must exist a vertex $v \in R$ such that $|\Gamma(S) \cap S| = 1$.

Now, take any $x \in C(G)$, and let $S \subset L$ be the support of x. We want to prove that $Hx = 0$, where H is the $m \times n$ parity check matrix of the code $C(G)$. Suppose that $|S| \leqslant d$, then by what we proved previously let $v \in R$ be the vertex such that $|\Gamma(v) \cap S| = 1$, hence the v-th coordinate of Hx is not zero.

In an attempt to make this more clear, we explain it differently. Observe that the whole v-th row of H has some 0's and 1's as entries, but there is only one entry 1 in the set S which is the support of x (this follows because of the unique neighbor result). Thus, the product of the v-th row and the vector x gives 1 as result, and therefore the entire product $Hx \neq 0$.

This means that every non-zero codeword $x \in C(G)$ has weight greater than d, and hence the weight of the code is also greater than d. □

We prove what is needed to show, that the codes from the bipartite graphs are efficient, i.e. that the codes can be decoded in polynomial time, in fact, linear time.

Consider the iterative decoding algorithm that upon receiving the input n-bit string y, as long as there exists a variable such that most of its neighbors constraints are not satisfied, it changes the entry of said variable. Explained differently, given $x \notin C$, we change its i-th entry granted that the Hamming weights satisfy $w(H(x + e_i)) < w(Hx)$, where H is the parity check matrix of G. This algorithm is known as the **belief propagation** algorithm.

Theorem 7.3.

Let G be a k-left-regular bipartite graph in which $L(G, d) > \frac{3}{4}k$. Let y be an n-bit string whose distance from a codeword x is at most $d/2$. Then a repeated application to y of the belief propagation algorithm, will return x after a linear number of iterations.

Proof.

Let $y^{(i)}$ denote the vector obtained after i iterations of the algorithm, and let $y = y^{(0)}$. Also, let A_i be the set of errors at step i, meaning that

$$A_i = \{v \in V(G) \ : \ y_v^{(i)} \neq x_v\}.$$

Hence, we want to prove that A_t is empty for $t = O(n)$.

Assume that $A_i \neq \emptyset$ and $|A_i| \leqslant d$, and consider a partition of $\Gamma(A_i)$ into sets of satisfied neighbors S_i and unsatisfied neighbors U_i. We will denote them S and U if it is clear from context. Note that U is the support of $A_i \ y^{(i)}$.

Clearly $|S| + |U| = |\Gamma(A_i)|$, hence from our assumptions on A_i and the left vertex expansion rate we obtain

$$|S| + |U| > \frac{3}{4}k\ |A_i|. \tag{7.1}$$

Now we count the edges between A_i and $\Gamma(A_i)$, which are $k|A_i|$. We claim that among them there are at least $|U|$ edges leaving U and at least $2|S|$ edges leaving S.

For $|U|$ it is clear since being part of the set of neighbors, at least there must be one edge for each vertex in the set. For $|S|$ we actually prove that there must be an even number of edges from S to A_i. Note that for a vertex $v \in S$ we have that

$$(Hy^{(i)})_v = 0,$$

because v is in the set of satisfied neighbors. Then given that A_i is the support of $Hy^{(0)}$ we obtain that

$$A_i \cdot y^{(i)} = 0,$$

which implies that there must be an even number of neighbors of v in A_i. Thus, there are in total at least $2|S|$ edges leaving S. This gives the inequality

$$|U| + 2|S| \leqslant k\ |A_i|. \tag{7.2}$$

Subtracting (7.2) from twice (7.1) gives that $|U| > \frac{1}{2}k\ |A_i|$. This means that there is a vertex (entry) of H with more than $k/2$ unsatisfied neighbors. We change the entry at that index and this switches all its neighbors from S to U and vice versa, but because there are more of such neighbors in U we obtain that $|H(y^{(i)}+e_i)| < |H(y^{(i)})|$.

Ultimately, this implies that $|U|$ decreases with every iteration of the belief propagation algorithm. Hence, if the distance from $y^{(i)}$ to x does not exceed d at any time, then the algorithm will halt at x after a linear number of iterations.

To finish the proof we must show that $|A_i|$ never exceeds d in size. Recall that $|A_0| \leqslant d/2$ by assumption, that gives $|U_0| \leqslant |\Gamma(A_0)| \leqslant kd/2$, and hence $|U_i| \leqslant kd/2$

for all i. Notice that after each step, $|A_i|$ increases or decreases in value in at most 1 entry. Therefore, if at some iteration $|A_l| > d$, then there exists $j < l$ such that $|A_j| = d|$, thus $|U_j| > kd/2$, which is a contradiction. □

7.2 Second Construction

Sipser and Spielman introduced an explicit family of asymptotically good and efficient linear codes in [6] and [7], using d-regular expander codes in the construction. Also, they provided an algorithm to remove a constant fraction of errors in the received encoded message. In this chapter we discuss a variation of their work, suggested by Gilles Zémor in [9], that corrects up to 12 times more errors without compromising the complexity of the algorithms. Gilles Zémor's improvement on the error correcting routine comes from the employment of bipartite Ramanujan graphs rather than general expander graphs.

We proceed to explain the construction of the binary code. Given a d-regular graph $G = (V, E)$, we will construct a code from it by considering an auxiliary bipartite graph G' which is based on the structure of G. Let $G' = (V \cup E, E')$, where $ab \in E'$ if and only if $a \in V$, $b \in E$ and there exists $c \in V$ such that $ac = b$. Therefore there exists edges only between the vertices of V and E, additionally, every vertex of V has exactly d edges to E, i.e. if we consider V to be the left set of the bipartite graph, then G' is a d-left-regular graph. Also, notice that every vertex of E is adjacent to exactly two vertices of V. We set an arbitrary labeling of the vertices of $E = \{1, \ldots, n\}$, and for any vertex $v \in V$ define $v(1), \ldots, v(d)$ to be some ordering of the d vertices of E which are adjacent to v. Then, let C_0 be a linear code of length $n_0 = d$, redundancy r_0, and minimum distance d_0.

Finally, we define the new code $C \subset \{0,1\}^n$ as the set of binary vectors $x =$

$(x_1, \ldots, x_n)$ such that for every vertex $v \in V$ we have that the smaller vector $(x_{v(1)}, \ldots, x_{v(d)})$ is a codeword in C_0. We use the notation (G, C_0) for the code C constructed from G in this way. We observe that such a code is not unique, the code obtained depends on the taken ordering of the edges $v(1), \ldots, v(d)$ given at each vertex v.

We have that the dimension of the constructed code C is at least $n(1 - 2r_0/n_0)$, where r_0 is the redundancy of the code C_0. Also, we have that its distance is at least $n\delta_0^2(1 - \varepsilon)$, where $\delta_0 = d_0/n_0$ is the relative minimum distance of C_0 and ε depends only of d_0, d, and $\lambda_2(G)$. Even more, if $\lambda_2/d_0 \to 0$, then we get also $\varepsilon \to 0$.

Now, we consider Ramanujan graphs, which are constructive families of graphs, with arbitrarily number of vertices for fixed degrees, and satisfy $\lambda_2 \leqslant 2\sqrt{d-1}$. More about Ramanujan graphs can be found in [5]. Hence if we choose our regular graph G to be a Ramanujan graph, then for a large enough d we obtain that $\lambda_2/d_0 \leqslant 2\sqrt{d-1}/(d \cdot d_0) \to 0$, thus we can make ε arbitrarily small and acquire asymptotically good (G, C_0)-codes. We will consider only bipartite Ramanujan graphs in the decoding algorithm for (G, C_0), however, this should not be a problem because there are plenty of known constructions.

For the decoding algorithm, let $G = (L, R, E)$ be a d-regular bipartite graph, and assume that $|L| = |R| = l$. This means that the graph has $n = dl$ edges, and therefore the length of a code (G, C_0) would be precisely n. Then, for any vertex v of G the subset of edges incident to v is

$$E_v = \{v(1), \ldots, v(d)\}.$$

In this manner, given that G is bipartite, we obtain two partitions of the edge set

$$E = \bigcup_{v \in L} E_v = \bigcup_{v \in R} E_v.$$

Let $x \in \{0, 1\}^n$ be the received vector. The first iteration of the algorithm consists of applying complete decoding in C_0 for the code induced by E_v for every $v \in L$. This can be done because, since the graph G is bipartite, the subsets of edges E_v for all v in L are disjoint. Thus, we are replacing the vector $(x_{v(1)}, \ldots, x_{v(d)})$ for one of the closest codewords in C_0, this for all vectors v in L. The iteration yields a new vector, say y, and for the second iteration we apply the same decoding but now for the partition induced by the set R. Afterwards we continue repeating this whole process, alternating between the decoding on the partitions induced by L and R.

We remark that this could not be done in the decoding algorithm for general d-regular graphs, the decoding would be done in all edges of the graphs by an argument that an edge bit would be changed if at least one of its related vertices thought it should be changed, more details can be found in [8]. Thus, the disjoint partition of the edge set is very important, which can additionally allow a parallel decoding at each step. When replacing for one of the closest codewords in C_0 it could be that the choice is not unique, and at this step we are not concerned on the complexity of the decoding in C_0 since the same small code C_0 will be used regardless of the size of the graph G, hence in this sense the complexity of such decoding is constant. Using the same argument, from a theoretical point of view one should not worry about finding a code C_0, once can choose the best already known ones.

We are now interested in a sufficient condition for the convergence of the algorithm. We will prove that, under certain conditions, if the weight of the error vector is less than $\alpha n \delta_0(\delta_0/2 - \lambda_2/d)/2$ for some $\alpha < 1$, then the previous decoding routine will converge to the initial codeword which was originally sent. Before we are able to prove such a theorem we are in need of proving the following two lemmas, in which for simplicity the average degree $\overline{\deg}_{G_{S\cup T}}$ will be denoted as $\overline{\deg}_{ST}$.

Lemma 7.4.

Let $G = (L, R, E)$ be a d-regular bipartite graph, with $|L| = |R| = l$, and let $S \subset L$ and $T \subset R$. Then the average degree $\overline{\deg}_{ST}$ of the induced subgraph $G_{S\cup T}$ satisfies

$$\overline{\deg}_{ST} \leqslant \frac{2|S||T|}{|S|+|T|}\frac{d}{l} + \lambda_2 - \frac{\lambda_2}{l}\frac{|S|^2+|T|^2}{|S|+|T|}.$$

Proof.

Let $A = A(G)$ be the $2l \times 2l$ adjacency matrix of the bipartite graph G, and let X_{ST} be the column vector of length $2l$ such that every coordinate indexed by a vertex of S or T equals 1 and is 0 otherwise, i.e. X_{ST} is the vector $1_S + 1_T$. We can deduce that

$$X_{ST}^{\mathrm{t}} A X_{XT} = \sum_{v \in S \cup T} \deg_{G_{S\cup T}}(v), \tag{7.3}$$

where $\deg_{G_{S\cup T}}(v)$ refers to the degree of the vertex v within the induced subgraph $G_{S\cup T}$. Additionally, let j be the all-one vector, and let $k = 1_L - 1_R$. Notice that j and k are eigenvectors of A associated to the eigenvalues d and $-d$, respectively. We clearly have $j \cdot j = 2l$, while $k \cdot k = |L| + |R| = 2l$, and $j \cdot k = |L| - |R| = 0$. At last, define Y_{ST} as

$$Y_{ST} = X_{ST} - \frac{|S| + |T|}{2l} j - \frac{|S| - |T|}{2l} k.$$

Observing that $X_{ST} \cdot j = |S| + |T|$ and $X_{ST} \cdot k = |S| - |T|$, it should be simple to compute from the previous relations that Y_{ST} is orthogonal to both j and k. Since the eigenspaces of A are orthogonal we can infer

$$X_{ST}^{\mathrm{t}} A X_{ST} = d \left(\frac{|S| + |T|}{2l} \right)^2 j \cdot j - d \left(\frac{|S| - |T|}{2l} \right)^2 k \cdot k + Y_{ST}^{\mathrm{t}} A Y_{ST},$$

which using the before mentioned relations is reduced to

$$X_{ST}^{\mathrm{t}} A X_{ST} = Y_{ST}^{\mathrm{t}} A Y_{ST} + 2d \frac{|S||T|}{l}.$$

Since Y_{ST} is orthogonal to j and the eigenspace associated to it is of dimension one because G is connected, then we obtain that $Y_{ST}^{\mathrm{t}} A Y_{ST} \leqslant \lambda_2 \|Y_{ST}\|^2$, and this together with equation (7.3) yields

$$\overline{\deg}_{ST}(|S| + |T|) \leqslant \lambda_2 \|Y_{ST}\|^2 + 2d \frac{|S||T|}{l}, \tag{7.4}$$

where $\overline{\deg}_{ST}$ is the average degree in the induced subgraph. Now, observe that Y_{ST} has $|S|$ coordinates equal to $1-|S|/l$, $|T|$ of them equal to $1-|T|/l$, $l-|S|$ equal to $-|S|/l$, and $l-|T|$ equal to $-|T|/l$, from which we can derive

$$\|Y_{ST}\|^2 = |S| + |T| - \frac{|S|^2+|T|^2}{l}.$$

This last equation combined with (7.4) proves the Lemma. □

Lemma 7.5.

Suppose that $d_0 \geqslant 3\lambda_2$. Let S be a subset of vertices of L, T a subset of vertices of R, and Y a subset of edges of E. If the following conditions are satisfied

i) $|S| \leqslant \alpha l \left(\frac{\delta_0}{2} - \frac{\lambda_2}{d} \right)$ *for some $\alpha < 1$.*

ii) Every edge of Y has one of its endpoints in S.

iii) Every vertex of T is incident to at least $d_0/2$ edges of Y,

then

$$|T| \leqslant \frac{1}{2-\alpha}|S|.$$

Proof.

Let $W \subset Y$ consist of those edges in Y that have one endpoint in T. Then, because of condition (ii), W is a subset of the set of edges $E(S \cup T)$. This implies that $E(S,T) \geqslant |W|$, and therefore the vertices in $G_{S\cup T}$ have an average degree

$$\overline{\deg}_{ST} \geqslant \frac{2|W|}{|S|+|T|}.$$

We observe that (iii) implies $|T|d_0/2 \leqslant |W|$, and combined with Lemma 7.4 gives

$$\frac{|T|d_0}{|S|+|T|} \leqslant \frac{2|W|}{|S|+|T|} \leqslant \overline{\deg}_{ST} \leqslant \frac{2|S||T|}{|S|+|T|}\frac{d}{l} + \lambda_2.$$

Applying assumption (iii), using that $\delta_0 d = \delta_0 n_0 = d_0$, and then solving the inequality for $|T|$ yields

$$|T| \leqslant \frac{\lambda_2}{d_0(1-\alpha)+\lambda_2(2\alpha-1)}|S|.$$

Finally, using the initial condition $d_0 \geqslant 3\lambda_2$ we obtain

$$|T| \leqslant \frac{\lambda_2}{3\lambda_2(1-\alpha)+\lambda_2(2\alpha-1)}|S| \leqslant \frac{1}{2-\alpha}|S|,$$

which is what we wanted to prove. □

Now that we have proved the previous lemmas we can prove the following theorem by Gilles Zémor, which gives a sufficient condition on the number of corrupted bits of the received vector x in order for the decoding routine to converge and return the original codeword, i.e. the algorithm corrects all reception errors.

Theorem 7.6.

Suppose $d_0 \geqslant 3\lambda_2$. If the weight of the error vector of x satisfies

$$|x| \;\leqslant\; \alpha n \cdot \frac{\delta_0}{2}\left(\frac{\delta_0}{2} - \frac{\lambda_2}{d}\right) \tag{7.5}$$

for some $\alpha < 1$, then the decoding algorithm converges to the initial codeword in a number of steps logarithmic in n.

Proof.

We assume, without loss of generality, that the original uncorrupted codeword is the zero codeword. We identify the error vector x with the set of erroneous edges

$X = \{i : x_i = 1\}$, i.e. X is the support of x. Let y be the vector obtained after one iteration of the decoding algorithm, and analogously let Y be its support. Similarly with z for the next iteration after that, and the set of edges Z.

Notice that for any given vertex $v \in L$, if such a vertex is incident to less than $d_0/2$ of the corrupted edges in X, then the local decoding in C_0 will erase all those errors, hence $E_v \cap Y = \emptyset$. Therefore, if we make $S = \{v \in L : E_v \cap Y \neq \emptyset\}$, it follows that

$$v \in S \text{ implies } |E_v \cap X| \geqslant d_0/2. \tag{7.6}$$

Analogously for $T \subset R$ such that $E_v \cap Z \neq \emptyset$, we obtain

$$v \in T \text{ implies } |E_v \cap Y| \geqslant d_0/2. \tag{7.7}$$

We show that S, T and Y satisfy the conditions of Lemma 7.5. From (7.6), since E_v forms a partition of the edges, we obtain that at least $|X| \geqslant |S| d_0/2$, which in conjunction with (7.5) gives

$$\frac{d_0}{2}|S| \leqslant |X| = |x| \leqslant \alpha n \cdot \frac{\delta_0}{2}\left(\frac{\delta_0}{2} - \frac{\lambda_2}{d}\right).$$

Since $d_0 = \delta_0 n_0$, $n_0 = d$, and $n = dl$, the previous inequality implies that $|S|$ satisfies the first requirement in Lemma 7.5. Then, the second condition on Lemma 7.5 is satisfied by the definition of S, while the third comes directly as a result of (7.7). All conditions of the Lemma are satisfied and therefore we have $|T| \leqslant 1/(2-\alpha)|S|$.

Let X_i be the set of erroneous edges after the decoding in step i, and let S_i the set of vertices defined as the set $\{v \in L : E_v \cap x_{i+1} \neq \emptyset\}$ if i is even, or as the set $\{v \in R : E_v \cap x_{i+1} \neq \emptyset\}$ if i is odd. With this definitions $S = S_0$ and $T = S_1$, hence he have already proved that $|S_1| \leqslant \beta|S|$, where $\beta = 1/(2-\alpha)$. This means that

$$|S_1| \leqslant \alpha\beta\, n\left(\frac{\delta_0}{2} - \frac{\lambda_2}{d}\right),$$

where clearly $\alpha_1 = \alpha\beta < 1$, and so S_1 also satisfies condition (i) of Lemma 7.5. The second and third conditions in the Lemma are always satisfied directly from the definition of the sets and the corresponding analogous to (7.7). Inductively we obtain that $|S_i| \leqslant \beta^i |S|$, and since $\beta < 1$ at some point we will obtain $S_j = \emptyset$, it is then that $X_{i+1} = \emptyset$ and the decoding algorithm will halt. □

It is worth noting that in the proof we showed the sets S_i decrease in size with each iteration of the decoding, however, the weight of the error vector does not necessarily decrease as well. Also, it is only at the first iteration that we require the condition (7.5) to prove the first requirement of Lemma 7.5, thereafter it is deduced from the properties of the previous set S_i.

Chapter 8

Conclusions

We briefly sum up how constructions of error correcting codes has been reached in this work.

- Firstly we introduced all definitions in Graph Theory that would later be used. We introduced the link between graphs and linear algebra, which allowed us to study properties of graphs based on their eigenvalues.

- We then introduced the idea of edge expansion, and from it we defined expander graphs. Shortly after we proved a lower and upper bound on the edge expansion rate, which allows us to determine if a family of graphs are expanders based only on their spectral gap.

- We presented the first expander graphs whose construction was explicit, the Margulis expander graphs, and we proved that they form indeed a family of expander graphs.

- We moved on to give an introduction into the field of Coding Theory. We introduced linear codes, and gave some asymptotic bounds on the rate and distance of a code.

- At this point we proceeded to give two explicit construction of error correcting codes induced by expander graphs. The first of them based on bipartite expander graphs, while the second utilized general expander graphs and a small linear code.

- The second construction was further improved by Gilles Zémor and we presented his proposal, which took advantage of properties of the Ramanujan graphs. We gave and proved a sufficient condition for which the decoding algorithm corrected all errors.

- Noticing that the induced code $C = (G, C_0)$ is not unique due to the arbitrary choice of the ordering of the edges incident to a given vertex v for the small code C_0, one could wonder if a careful choice of the ordering can give rise to a better code C. One can also wonder if a different structure could be put in place instead of a linear code.

Appendix

We will very briefly introduce some definitions and mention some results in harmonic analysis, which are used to further study properties of graphs and their associated eigenvalues, this theory will come in handy specially when proving that the Margulis graphs form a family of expander graphs. Let $\mathcal{F}$ be the collection of all complex functions on a group H, with inner product

$$< f, g >= \sum_{x \in H} f(x)\overline{g(x)}.$$

A **character** of a group H is a homomorphism $\chi : H \to \mathbb{C}^*$, i.e. $\chi(gh) = \chi(g) \cdot \chi(h)$ for all $g, h \in H$. When H is Abelian we denote its group operation by $+$.

The discrete Fourier transform of a complex function f is

$$\hat{f}(x) =< f, \chi_x >= \sum_{y \in H} \overline{f(y)}\chi_x(y).$$

We can express functions in $\mathcal{F}$ as linear combinations of characters.

Theorem 8.1.

Every finite Abelian group H has $|H|$ distinct characters which can be indexed as $\chi_{x x \in H}$. These characters form an orthonormal basis of $\mathcal{F}$. Then every $f : H \to \mathbb{C}$ can be uniquely expressed as

$$f = \sum_{x \in H} \hat{f}(x) \cdot \chi_x.$$

We proceed to mention without proof basic properties of the Fourier transform for the particular case when the group H is $\mathbb{Z}_n^2$.

Remark 8.2. *For $f, g \in \mathcal{F}$ we have*

- $\sum_{a \in H} f(a) = 0$ *if and only if* $\hat{f}(0) = 0$.
- $< f, g >= \frac{1}{n^2} < \hat{f}, \hat{g} >$.
- *Parseval's identity*

$$\sum_{a \in H} |f(a)|^2 = \frac{1}{n^2} \sum_{a \in H} |\hat{f}(a)|^2.$$

- *The inverse formula*

$$f(a) = \frac{1}{n^2} \sum_{b \in H} \hat{f}(b) \cdot w^{-<a,b>}.$$

- *If A is a non-singular 2×2 matrix over $\mathbb{Z}_n$, $b \in H$ and $g(x) = f(Ax + b)$, then*

$$\hat{g}(y) = w^{-<A^{-1}b,y>} \hat{f}((A^{-1})^T y).$$

Bibliography

[1] J.A. Bondy, U.S.R. Murty (2008), *Graph Theory.* Springer

[2] Y. Bilu, N. Linial. (2006), *Lifts, Discrepancy and Nearly Optimal Spectral Gaps.* Combinatorica. Vol 26, Number 5: 495-519

[3] F. Harary (1969), *Graph Theory.* Addison-Wesley

[4] S. Hoory, N. Linial, A. Wigderson (2006), *Expander Graphs and their Applications.* Bulletin of the American Mathematical Society. Vol 43, Number 4: 439-561

[5] A. Lubotsky, R. Phipips, P. Sarnak. (1988), *Ramanujan graphs.* Combinatorica. Vol 8, Number 3: 261-277

[6] M. Sipser, D.A. Spielman (1996), *Expander Codes.* IEEE Transactions on Information Theory. Vol 42, Number 6: 1710-1722

[7] D.A. Spielman (1996), *Linear-Time Encodable and Decodable Error-Correcting Codes.* IEEE Transactions on Information Theory. Vol 42, Number 6: 1723-1731

[8] D.A. Spielman (1996), *Constructing Error-Correcting Codes from Expander Graphs.* IMA Volumes in Mathematics and its Applications. Vol 109.

[9] G. Zémor (2001), *On Expander Codes.* IEEE Transactions on Information Theory. Vol 47, Number 2: 835-837