Chapter 5

Joint Iterative Decoding of LDPC Codes and Channels with Memory

5.1 Introduction

Sequences of irregular low-density parity-check (LDPC) codes that achieve the capacity of the binary erasure channel (BEC) under iterative decoding were first constructed by Luby, *et al.* in [11]. This was followed by the work of Chung, *et al.*, *which* provided evidence suggesting that sequences of iteratively decoded LDPC codes can also achieve the channel capacity of the binary-input additive white Gaussian noise (AWGN) channel [3]. Since then, density evolution (DE) [13] has been used to optimize irregular LDPC codes for a variety of memoryless channels (e.g., [6]), and the results suggest, for each channel, that sequences of iteratively decoded LDPC codes can indeed achieve the channel capacity. In fact, the discovery of a channel whose capacity cannot be approached by LDPC codes would be more surprising than a proof that iteratively decoded LDPC codes can achieve the capacity of any binary-input symmetric channel (BISC).

The idea of decoding a code transmitted over a channel with memory via iteration was first introduced by Douillard, *et al.* in the context of turbo codes and is known as *turbo equalization* [4]. This approach can also be generalized to LDPC codes by constructing one large graph which represents the constraints of both the channel and the code. This idea is also referred to as *joint iterative decoding*, and was investigated for partial-response channels by

Figure 5.1.1: Block diagram of the system.

Kurkoski, Siegel, and Wolf in [9].

Until recently, it was difficult to compare the performance of turbo equalization with channel capacity because the binary-input capacity of the channel was unknown. Recently, a new method has gained acceptance for estimating the achievable information rates of finite state channels (FSCs) [1][12][1], and a number of authors have begun designing LDPC based coding schemes which approach the achievable information rates of these channels [8][12][21]. As is the case with DE for general BISCs, the evaluation of code thresholds and the optimization of these thresholds is done numerically. For FSCs, the analysis of this system is quite complex because the BCJR algorithm [2] is used to decode the channel.

Since the capacity of a channel with memory is generally not achievable via equiprobable signaling, one can instead aim for the symmetric information rate (SIR) of the channel. The SIR is defined as the maximum information rate achievable via random coding with equiprobable input symbols. Since linear codes use all inputs equiprobably, the SIR is also the maximum rate directly achievable with linear codes. In this chapter, we introduce a class of channels with memory, which we refer to as generalized erasure channels (GECs). For these channels, we show that DE can be done analytically for the joint iterative decoding of irregular LDPC codes and the channel. This allows us to construct sequences of LDPC degree distributions which appear to achieve the SIR using iterative decoding. As an example, we focus on the dicode erasure channel (DEC), which is simply a binary-input channel with a linear response of $1-D$ and erasure noise.

In Section 5.2, we introduce the basic components of our system. This includes the joint iterative decoder, GECs and the DEC, and irregular LDPC codes. In Section 5.3, we derive a single parameter recursion for the DE of the joint iterative decoder which allows us to give necessary and sufficient conditions for decoder convergence. These conditions are also used to construct code sequences which appear to achieve the SIR. In Section 5.4, we discuss extensions

[1]This method was also introduced by Sharma and Singh in [14]. However, it appears that most of the other results in their paper, based on regenerative theory, are actually incorrect. A correct analytical treatment can be found in Section 4.4.4.

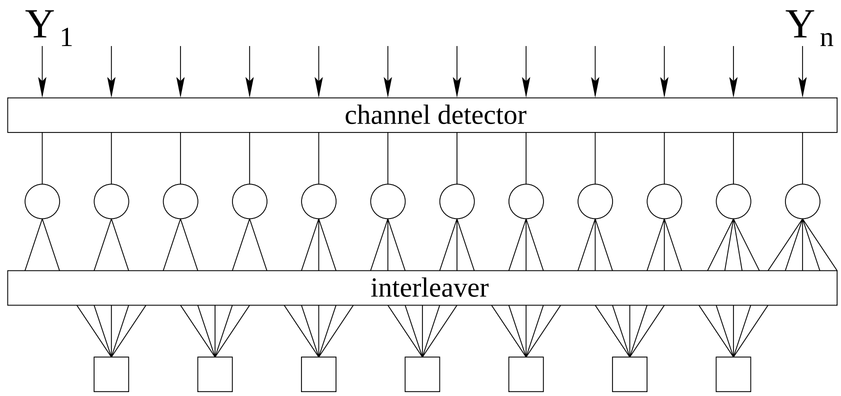

Figure 5.2.1: Gallager-Tanner-Wiberg graph of the joint iterative decoder.

to arbitrary GECs and describe a practical optimization technique based on linear programming. Finally, we offer some concluding remarks in Section 5.5.

5.2 System Model

5.2.1 Description

The system we consider is fairly standard for the joint iterative decoding of an LDPC code and a channel with memory. Equiprobable information bits, $\mathbf{U} = U_1, \ldots, U_k$, are encoded into an LDPC codeword, $\mathbf{X} = X_1, \ldots, X_n$, which is observed through a GEC as the output vector, $\mathbf{Y} = Y_1, \ldots, Y_n$. The decoder consists of an *a posteriori probability* (APP) detector matched to the channel and an LDPC decoder. The first half of decoding iteration i entails running the channel detector on $\mathbf{Y}$ using the *a priori* information from the LDPC code. The second half of decoding iteration i corresponds to executing one LDPC iteration using internal edge messages from the previous iteration and the channel detector output. Figure 5.1.1 shows the block diagram of the system, and Figure 5.2.1 shows the Gallager-Tanner-Wiberg (GTW) graph of the joint iterative decoder.

5.2.2 The Generalized Erasure Channel

Since the messages passed around the GTW graph of the joint decoder are all log-likelihood ratios (LLRs), DE involves tracking the evolution of the distribution of LLR messages passed around the decoder. Let L be a random variable representing a randomly chosen LLR at the output of the channel decoder. If the distribution of L is supported on the set $\{-\infty, 0, \infty\}$ and $Pr(L = -\infty) = Pr(L = \infty)$, then we refer to it as a *symmetric erasure distribution*. Such distributions are one dimensional, and are completely defined by the erasure probability $Pr(L = 0)$. Our closed form analysis of this system requires that all the densities involved in DE are symmetric erasure distributions.

Definition 5.2.1. A *generalized erasure channel* (GEC) is any channel which satisfies the following condition for i.i.d. equiprobable inputs. The LLR distribution at the output of the channel detector is a symmetric erasure distribution whenever the *a priori* LLR distribution is a symmetric erasure distribution.

This allows DE of the joint iterative decoder to be represented by a single parameter recursion. Let $f(x)$ be a function which maps the erasure probability of the *a priori* LLR distribution, x, to the erasure probability at the output of the detector. The effect of the channel on the DE depends only on $f(x)$, which we refer to as the *erasure transfer function* (ETF) of the GEC. This function is very similar to the mutual information transfer function, $T(I)$, used by the EXIT chart analysis of ten Brink [18]. Since the mutual information of a BEC with erasure probability x is $1 - x$, the mutual information transfer function and $f(x)$ are linked by the identity, $T(I) = 1 - f(1 - I)$.

A remarkable connection between the SIR of a channel, I_s, and its mutual information transfer function was also introduced by ten Brink in [19]. This result requires that $T(I)$ is computed using a symmetric erasure distribution as the *a priori* LLR distribution. A clever application of the chain rule for mutual information shows that

$$\lim_{n\to\infty} \frac{1}{n} I(X_1, \ldots, X_n; Y_1, \ldots, Y_n) = \int_0^1 T(I)dI.$$

Assuming the input process is i.i.d. and equiprobable makes the LHS equal the SIR, and using $T(I) = 1 - f(1 - I)$ allows us to simplify this expression to

$$I_s = \int_0^1 T(I)dI = 1 - \int_0^1 f(x)dx. \tag{5.2.1}$$

Previously, we saw that $f(x)$ completely characterizes the DE properties of a GEC, and now we see that it can also be used to compute the SIR.

5.2.3 The Dicode Erasure Channel

The dicode erasure channel (DEC) is a binary-input channel based on the $1-D$ linear intersymbol-interference (ISI) dicode channel used in magnetic recording. Essentially, the output of the dicode channel $(+1, 0, -1)$ is erased with probability ϵ and transmitted perfectly with probability $1-\epsilon$. The precoded DEC is essentially the same, except that the input bits are differentially encoded prior to transmission. This modification simply changes the input labeling of the channel state diagram. The state diagram of the dicode channel is shown with and without precoding in Figure 5.2.2.

The simplicity of the DEC allows the BCJR algorithm for the channel to be analyzed in closed form. The method is similar to the exact analysis of turbo codes on the BEC [20], and the result shows that the DEC is indeed a GEC. Leaving the details to Appendix 5A, we state the ETFs for the DEC with and without precoding. If there is no precoding and the outputs of the DEC are erased with probability ϵ, then the ETF of the channel detector is

$$f(x) = \frac{4\epsilon^2}{(2 - x(1-\epsilon))^2}. \tag{5.2.2}$$

On the other hand, using a precoder changes this function to

$$f(x) = \frac{4\epsilon^2 x\,(1 - \epsilon(1-x))}{(1-\epsilon(1-2x))^2}. \tag{5.2.3}$$

Analyzing only the forward recursion of the BCJR algorithm allows one to compute the SIR of the DEC, and the result, which was computed in Section 4.7.1, is given by

$$I_s(\epsilon) = 1 - \frac{2\epsilon^2}{1+\epsilon}.$$

It is easy to verify that one can also get this expression for the SIR from either (5.2.2) or (5.2.3) by applying (5.2.1).

5.2.4 Irregular LDPC Codes

Irregular LDPC codes are a generalization of Gallager's LDPC codes [5] that have been shown to perform remarkably well under iterative decoding [13]. They are probably best understood by considering their graphical representation as a bipartite graph, which is shown at bottom of Figure 5.2.1. Iterative decoding is performed by passing messages along the edges of this graph, and the evolution of these messages can be tracked using DE. In general, when we

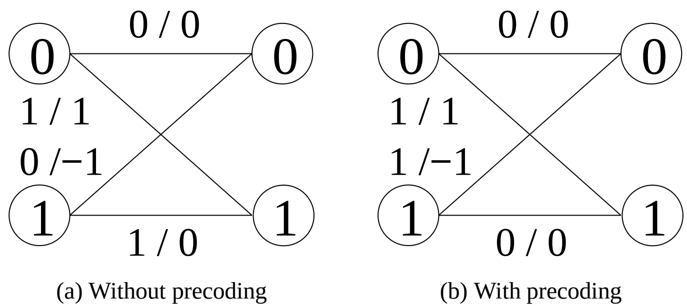

Figure 5.2.2: The state diagram of the dicode channel with and without precoding.

speak of an LDPC code we are referring to the ensemble of codes formed by picking a random bipartite graph with the proper degree structure.

For channels with memory, the standard DE assumption of channel symmetry may not hold. Essentially, this means that DE can only be applied to one codeword at a time. In [7], the i.i.d. channel adaptor is introduced as a conceptual device which ensures the symmetry of any channel. If the outer code is a linear code, then this approach is identical to choosing a random coset and treating it as part of the channel. In this work, we employ the i.i.d. channel adaptor approach by assuming that the choice of a random coset vector is embedded in the channel.

The degree distribution of an irregular LDPC code can be viewed either from the edge or node perspective, and the results of this chapter are simplified by using both perspectives. Let $\lambda(x)$ be a polynomial defined by $\lambda(x) = \sum_{\nu \geq 1} \lambda_\nu x^{\nu-1}$, where λ_ν is the fraction of edges attached to a bit node of degree ν. Likewise, let $\rho(x)$ be a polynomial defined by $\rho(x) = \sum_{\nu \geq 1} \rho_\nu x^{\nu-1}$, where ρ_ν is the fraction of edges attached to a check node of degree ν. We refer to $\lambda(x)$ and $\rho(x)$ as the bit and check degree distribution from the edge perspective. Let $L(x)$ be a polynomial defined by $L(x) = \sum_{\nu \geq 1} L_\nu x^\nu$, where L_ν is the fraction of bit nodes with degree ν. Let $R(x)$ be a polynomial defined by $R(x) = \sum_{\nu \geq 1} R_\nu x^\nu$, where R_ν is the fraction of check nodes with degree ν. We refer to $L(x)$ and $R(x)$ as the bit and check degree distributions from the node perspective. The coefficients of all these polynomials represent a fraction of some whole, and that means that $\lambda(1) = \rho(1) = L(1) = R(1) = 1$. Using the definitions of $L(x)$ and $R(x)$, it is also easy to verify that $L(0) = 0$ and $R(0) = 0$. Finally, we note that the possibility of bit and check nodes with degree 1 was intentionally included, so we cannot assume that $\lambda(0) = 0$ or $\rho(0) = 0$.

The average bit degree, a_L, and the average check degree, a_R, are easily computed to be $a_L = \sum_{\nu \geq 1} L_\nu \nu = L'(1)$ and $a_R = \sum_{\nu \geq 1} R_\nu \nu = R'(1)$. One can also switch from the bit to edge perspective by noting that each node of degree ν contributes ν edges to the edge perspective. Counting from the edge perspective and normalizing gives

$$\lambda(x) = \frac{\sum_{\nu \geq 1} L_\nu \nu x^{\nu-1}}{\sum_{\nu \geq 1} L_\nu \nu} = \frac{L'(x)}{a_L} \tag{5.2.4}$$

and

$$\rho(x) = \frac{\sum_{\nu \geq 1} R_\nu \nu x^{\nu-1}}{\sum_{\nu \geq 1} R_\nu \nu} = \frac{R'(x)}{a_R}. \tag{5.2.5}$$

Changing from the edge to bit perspective can be accomplished by integrating both sides of these expressions. This also gives the alternative formulas, $a_L = 1/\int_0^1 \lambda(t)dt$ and $a_R = 1/\int_0^1 \rho(t)dt$. Finally, we note that the rate of an irregular LDPC code is given by $R = 1 - a_L/a_R$.

Iterative decoding of irregular LDPC codes on the BEC, with erasure probability δ, was introduced by Luby *et al.* in [11] and refined in [10]. The recursion for the erasure probability out of the bit nodes is given by

$$x_{i+1} = \delta\lambda\left(1 - \rho(1 - x_i)\right), \tag{5.2.6}$$

while the dual recursion for edges out of the check nodes is given by

$$y_{i+1} = 1 - \rho\left(1 - \delta\lambda(y_i)\right). \tag{5.2.7}$$

Applying linear programming (LP) to these recursions allows one to maximize the code rate over one degree distribution while holding the other one fixed [11]. Although this type of alternating maximization can have convergence problems, it does provide a technique for optimizing degree distribution sequences which works well in practice.

5.3 Analysis of Joint Iterative Decoding

5.3.1 Single Parameter Recursion

Now, we consider a turbo equalization system which performs one channel iteration for each LDPC code iteration. The function, $f(x)$, gives the fraction of erasures produced by the extrinsic output of the channel decoder when the *a priori* erasure rate is x. The update equations for this system are almost identical to (5.2.6) and (5.2.7). The main difference is that the parameter δ now changes with iteration and is written as δ_i.

Consider the messages passed from the output of the check nodes to the input of the bit nodes, and let x be the fraction which are erased. Since any non-erasure message passed into a bit node gives perfect knowledge of the bit, the messages at the output of the bit node will only be erased only if all of the messages at the input to the bit node are erased. Therefore, the fraction of erased messages passed back from a bit node of degree ν to the check nodes is given by $\delta_i x^{\nu-1}$. Using this, it is easy to verify that the fraction of erased messages passed back from all the bit nodes to the check nodes is given by $\delta_i \sum_{\nu \geq 1} \lambda_\nu x^{\nu-1} = \delta_i \lambda(x)$.

There is also a fundamental difference between the messages passed from the bit nodes to the check nodes and the messages passed from the bit nodes to the channel detector. This difference is due to the fact that a degree ν bit node sends ν messages to the check nodes and only 1 message to the channel detector. The fraction of erased messages passed from a degree ν bit node to the channel detector is given by x^ν. Combining these two observations, shows that the fraction of erased messages passed from all of the bit nodes to the channel detector is $\sum_{\nu \geq 1} L_\nu x^\nu = L(x)$.

The recursion for the erasure probability out of the bit nodes is now given by

$$x_{i+1} = \delta_i \lambda \left(1 - \rho(1 - x_i)\right), \tag{5.3.1}$$

where $\delta_i = f\left(L\left(1 - \rho(1 - x_i)\right)\right)$ and $x_0 = f(1)$. Likewise, the dual recursion for edges out of the check nodes is now given by

$$y_{i+1} = 1 - \rho\left(1 - \delta_i \lambda(y_i)\right),$$

where $\delta_i = f\left(L(y_i)\right)$ and $y_0 = 1 - \rho\left(1 - f(1)\right)$.

5.3.2 Conditions for Convergence

Using the recursion (5.3.1), we can derive a necessary and sufficient condition for the erasure probability to converge to zero. This condition is typically written as a basic condition which must hold for $x \in (0, 1]$ and an auxiliary stability condition which simplifies the analysis at $x = 0$. The basic condition implies there are no fixed points in the iteration for $x \in (0, 1]$ and is given by

$$f\left(L\left(1 - \rho(1 - x)\right)\right) \lambda\left(1 - \rho(1 - x)\right) < x. \tag{5.3.2}$$

Verifying this condition numerically for very small x can be difficult, so we require instead that $x = 0$ is a stable fixed point of the recursion. This is equivalent to evaluating the derivative of (5.3.2) at $x = 0$, which gives the stability condition

$$\left(\lambda^2(0) f'(0) a_L + \lambda'(0) f(0)\right) \rho'(1) < 1. \tag{5.3.3}$$

The following facts make it easy to derive (5.3.3) from (5.3.2): $\rho(1) = 1$, $L(0) = 0$, and $L'(x) = a_L \lambda(x)$.

Now, we can use (5.3.2) and (5.3.3) to say something about the code properties required by various channels.

1. If the channel has $f(0) > 0$ and the code has $\lambda(0) > 0$, then $f(0)\lambda(0) > 0$ and (5.3.2) cannot hold near zero. This means that $\lambda(0) = 0$ is required for the satisfaction of (5.3.2), and this implies that the code cannot have degree 1 bit nodes (i.e., bits with very little code protection). In this case, the stability condition simplifies to $\lambda_2 f(0) \rho'(1) \le 1$.

2. If the channel has $f(0) = 0$, then degree 1 bit nodes do not cause this problem. In this case, the stability condition simplifies to $\lambda_1^2 f'(0) a_L \rho'(1) < 1$.

3. If the channel has $f(1) = 1$ and the code has $\rho(0) = 0$, then iteration cannot proceed beyond the fixed point at $x = 1$. It follows that the code must have $\rho(0) > 0$ to get the iteration started. In this case, the required degree 1 check nodes essentially act very much like pilot bits.

The next step is mapping (5.3.2) into an equivalent condition which is easier to manipulate. Consider the condition

$$f\left(L\left(1 - \rho(1 - q(x))\right)\right) \lambda\left(1 - \rho(1 - q(x))\right) < q(x),$$

for any $q(x)$ which is a one-to-one mapping from the interval $(0, 1]$ to the interval $(0, 1]$. This new condition is equivalent to the original condition (5.3.2). Choosing $q(x) = 1 - \rho^{-1}(1 - x)$ collapses the basic condition to

$$f\left(L(x)\right) \lambda(x) < q(x) \tag{5.3.4}$$

because $q^{-1}(x) = 1 - \rho(1 - x)$. Using (5.2.4), we can substitute for $\lambda(x)$ to get

$$f\left(L(x)\right) L'(x) < a_L q(x). \tag{5.3.5}$$

Integrating both sides of this inequality from 0 to x gives

$$F\left(L(x)\right) < a_L Q(x),$$

where $F(x) = \int_0^x f(t)dt$ and $Q(x) = \int_0^x q(t)dt$. We note that the function $F(x)$ is non-decreasing for $x \geq 0$ because the function $f(x)$ is non-negative for $x \geq 0$. This means that $F(x)$ is invertible for $x \geq 0$ and we can solve for $L(x)$ by writing

$$L(x) < F^{-1}\left(a_L Q(x)\right). \tag{5.3.6}$$

It appears that we now have a closed form condition involving only $L(x)$ which can be used to analyze the system. Unfortunately, this condition (5.3.6) does not imply (5.3.2) because the sequence of transformations is not reversible. Integrating both sides of an inequality preserves the inequality, but working backwards requires that we take the derivative of both sides. This does not, in general, preserve the inequality.

One way to work around this problem is to require that each step of the above derivation holds with equality. If we assume that (5.3.6) holds with equality, then we can takes its derivative and solve for $\lambda(x)$ to get

$$\lambda(x) = \frac{q(x)}{f\left(F^{-1}\left(a_L Q(x)\right)\right)}. \tag{5.3.7}$$

It is easy to verify this step using the facts that $\frac{d}{dx}F^{-1}(x) = 1/f\left(F^{-1}(x)\right)$, $Q'(x) = q(x)$, and $L'(x) = a_L \lambda(x)$. Finally, we note that (5.3.7) implies that the basic condition (5.3.2) holds with equality as well.

The following theorem relates these inequalities to the gap between the SIR and the code rate.

Theorem 5.3.1. *Consider any LDPC code ensemble, defined by the degree distributions $\lambda(x)$ and $\rho(x)$, which satisfies (5.3.2) for some GEC with ETF $f(x)$. The gap, Δ, between the rate of the LDPC code and the SIR of the channel, I_s, is given by*

$$\Delta = I_s - R = \int_0^1 g(x)dx,$$

where $g(x) = a_L q(x) - f\left(L(x)\right) L'(x)$ is the non-negative gap between the LHS and the RHS of (5.3.5).

Proof. Evaluating the integral gives

$$\int_0^1 g(x)dx = a_L\left[Q(1)-Q(0)\right] - \left[F\left(L(1)\right) - F\left(L(0)\right)\right],$$

where $Q(0)=0$, $F(L(0))=F(0)=0$, and $F(L(1))=F(1)=1-I_s$. We can compute $Q(1)$ using the geometric fact that

$$\int_0^1 \rho(x)dx + \int_0^1 \rho^{-1}(x)dx = 1,$$

and this gives

$$Q(1) = \int_0^1 \left(1-\rho^{-1}(1-x)\right)dx = \int_0^1 \left(1-\rho^{-1}(x)\right)dx = \int_0^1 \rho(x)dx = \frac{1}{a_R}.$$

Putting these together with the fact that $R = 1 - a_L/a_R$ gives

$$\int_0^1 g(x)dx = \frac{a_L}{a_R} - (1-I_s) = I_s - R.$$

□

5.3.3 Achieving the Symmetric Information Rate

Now, we consider sequences of irregular LDPC code ensembles which can be used to communicate reliably at rates arbitrarily close to the SIR. The code sequence is defined by the sequence of degree distributions $\left\{\rho^{(k)}(x),\lambda^{(k)}(x)\right\}_{k\geq 0}$ and its associated rate sequence $\{R_k\}_{k\geq 0}$ is given by $R_k = 1 - a_L^{(k)}/a_R^{(k)}$ via the results of Section 5.2.4. The main difficulty that we will encounter while using algebraic methods to define code sequences is that the implied degree distributions may not be non-negative and generally have infinite support. We say that a degree distribution is (i) *admissible* if its power series expansion about $x=0$ has only non-negative coefficients and (ii) *realizable* if it is a polynomial (i.e., finite degree) whose coefficients sum to one. We say that a sequence of degree distributions is *SIR achieving* if, for any $\epsilon > 0$, there exists an k_0 such that, for all $k > k_0$, the kth degree distribution is (i) realizable, (ii) satisfies (5.3.2), and (iii) has rate $R_k > I_s - \epsilon$.

The following corollary of Theorem 5.3.1 provides a necessary and sufficient condition for an SIR achieving sequence of degree distributions.

Corollary 5.3.2. *Consider any sequence of LDPC code ensembles, defined by the sequence* $\left\{\rho^{(k)}(x),\lambda^{(k)}(x)\right\}_{k\geq 0}$ *of realizable degree distributions, which satisfy (5.3.2) for some GEC* *with ETF $f(x)$. This sequence of codes is SIR achieving if and only if the associated sequence of rate gap functions, defined by*

$$g^{(k)}(x) = a_L^{(k)} \left[q^{(k)}(x) - f\left(L^{(k)}(x)\right) \lambda^{(k)}(x) \right],$$

converges to zero almost everywhere on $[0,1]$.

Proof. The definition of SIR achieving requires that the associated sequence of rate gaps, defined by $\Delta_k = \int_0^1 g^{(k)}(x)dx$, approaches zero. Since $g^{(k)}(x) > 0$ on $(0,1]$ by assumption, this requires that $\lim_{k\to\infty} g^{(k)}(x) = 0$ almost everywhere on $[0,1]$. □

Remark 5.3.3. For the BEC, Shokrollahi [16] showed that all sequences of capacity achieving codes obey a flatness condition which says that the sequence of non-negative gap functions implied by the basic condition, defined by

$$f\left(L^{(k)}\left(1-\rho^{(k)}(1-x)\right)\right)\lambda^{(k)}\left(1-\rho^{(k)}(1-x)\right) - x,$$

converges (along with all of its derivatives) to zero uniformly on $[0,1]$. We believe this can probably be extended to GECs under the assumption that the power series expansion of $f(x)$ about $x=0$ converges uniformly on $[0,1]$.

In general, we construct SIR achieving sequences by starting with a sequence of realizable check degree distributions $\left\{\rho^{(k)}(x)\right\}_{k\geq 0}$ and then using a slight variation of (5.3.7) to define a sequence of bit degree distributions $\left\{\widetilde{\lambda}^{(k)}(x)\right\}_{k\geq 0}$. If each bit degree distribution in this sequence is admissible with $\widetilde{\lambda}^{(k)}(1) > 1$, then we can form the sequence of realizable bit degree distributions $\left\{\lambda^{(k)}(x)\right\}_{k\geq 0}$ by truncating the power series of each $\widetilde{\lambda}^{(k)}(x)$ so that $\lambda^{(k)}(1) = 1$. Specifically, we generalize the notation of Section 5.2.4 and let $\lambda_i^{(k)} = \widetilde{\lambda}_i^{(k)}$ for $1 \leq i < N_k$, where N_k is the smallest integer such that $\sum_{i=1}^{N_k} \widetilde{\lambda}_i^{(k)} \geq 1$. The last term $\lambda_{N_k}^{(k)}$ is then chosen so that $\lambda^{(k)}(1) = 1$.

One problem with this method, which does not occur for the BEC [15], is that the truncation may cause the basic condition (5.3.2) to fail. To overcome this problem, we require the codes in sequence to satisfy the slightly stronger condition that

$$(1+\alpha_k) f\left(\widetilde{L}^{(k)}(x)\right) \widetilde{\lambda}^{(k)}(x) = q^{(k)}(x), \tag{5.3.8}$$

where $\widetilde{L}^{(k)}(x) = \int_0^x \widetilde{\lambda}^{(k)}(t)dt / \int_0^1 \widetilde{\lambda}^{(k)}(t)dt$ and $q^{(k)}(x) = 1 - \left(\rho^{(k)}\right)^{-1}(1-x)$. This is essentially the same as designing codes for a sequence of degraded channels given by $f^{(k)}(x) =$

$(1+\alpha_k)f(x)$. Adapting (5.3.6) to our system with equality gives

$$\widetilde{L}^{(k)}(x) = F^{-1}\left(\frac{1}{1+\alpha_k}\widetilde{a}_L^{(k)} Q^{(k)}(x)\right), \tag{5.3.9}$$

where $Q^{(k)}(x) = \int_0^x q^{(k)}(t)dt$. Requiring that $\widetilde{L}^{(k)}(1) = 1$ is the same as choosing $\widetilde{a}_L^{(k)}$ so that, without truncation, the code rate equals the SIR of the degraded channel. This gives

$$\widetilde{a}_L^{(k)} = a_R^{(k)}(1+\alpha_k)F(1), \tag{5.3.10}$$

because from (5.2.1) we have $I_s = 1 - F(1)$. Taking the derivative of (5.3.9) and substituting for $\widetilde{a}_L^{(k)}$ with (5.3.10) gives

$$\widetilde{\lambda}^{(k)}(x) = \frac{q^{(k)}(x)}{(1+\alpha_k)f\left(F^{-1}\left(F(1)a_R^{(k)}Q^{(k)}(x)\right)\right)}. \tag{5.3.11}$$

Notice that the channel only enters this equation via the expression $f\left(F^{-1}\left(F(1)x\right)\right)$ and that varying α_k really only changes the truncation point for $\lambda^{(k)}(x)$. Using the facts that $q^{(k)}(x) = 1$ and $Q^{(k)}(1) = 1/a_R^{(k)}$, we also note that

$$\widetilde{\lambda}^{(k)}(1) = \frac{1}{(1+\alpha_k)f(1)}. \tag{5.3.12}$$

This means that the truncation will work, for small enough α_k, as long as $f(1) < 1$.

Theorem 5.3.4. *Let $\left\{\rho^{(k)}(x)\right\}_{k\geq 0}$ be a sequence of realizable check degree distributions and let $\left\{\widetilde{\lambda}^{(k)}(x)\right\}_{k\geq 0}$ be the sequence of bit degree distributions given by (5.3.11). Suppose that (i) the first derivative of $f(x)$ is bounded on $[0,1]$ and $f(1) < 1$, (ii) each $\widetilde{\lambda}^{(k)}(x)$ given by (5.3.11) with $\alpha_k = 0$ is admissible, and (iii) the average check degree $a_R^{(k)}$ and maximum bit degree N_k satisfy $a_R^{(k)}/N_k \to 0$. Under these conditions, there exists a positive sequence $\{\alpha_k\}_{k\geq 0}$ such that the sequence of degree distributions $\left\{\rho^{(k)}(x), \lambda^{(k)}(x)\right\}_{k\geq 0}$ defined above is SIR achieving.*

Proof. We start by examining the effect of the power series truncation. This gives the the sandwich inequality

$$\widetilde{\lambda}^{(k)}(x) - \left(\widetilde{\lambda}^{(k)}(1) - 1\right)x^{N_k-1} \leq \lambda^{(k)}(x) < \widetilde{\lambda}^{(k)}(x), \tag{5.3.13}$$

where the LHS holds because $\left(\widetilde{\lambda}^{(k)}(1) - 1\right)x^{N_k-1}$ is an upper bound on the truncated terms and the RHS holds by truncation of positive terms. Now, consider the integral representation,

$L^{(k)}(x) = \int_0^x \lambda^{(k)}(t)dt / \int_0^1 \lambda^{(k)}(t)dt$, implied by integrating (5.2.4). Using this and (5.3.13), we get a sandwich inequality for $L^{(k)}(x)$ given by

$$\frac{\int_0^x \left(\widetilde{\lambda}^{(k)}(t) - \left(\widetilde{\lambda}^{(k)}(1) - 1\right) t^{N_k - 1}\right) dt}{\int_0^1 \widetilde{\lambda}^{(k)}(t)dt} < L^{(k)}(x) < \frac{\int_0^x \widetilde{\lambda}^{(k)}(t)dt}{\int_0^1 \left(\widetilde{\lambda}^{(k)}(t) - \left(\widetilde{\lambda}^{(k)}(1) - 1\right) t^{N_k - 1}\right) dt},$$

where $\lambda^{(k)}(x)$ is upper/lower bounded by the RHS/LHS of (5.3.13) respectively. Evaluating the integrals and rearranging terms reduces this to

$$\widetilde{L}^{(k)}(x) - \beta_k x^{N_k} < L^{(k)}(x) < \frac{1}{1 - \beta_k} \widetilde{L}^{(k)}(x), \tag{5.3.14}$$

where $\beta_k = \left(\widetilde{\lambda}^{(k)}(1) - 1\right) \widetilde{a}_L^{(k)} / N_k$. Using (5.3.10) and (5.3.12), we also see that

$$\beta_k = \left(\frac{1}{(1 + \alpha_k) f(1)} - 1\right) \frac{a_R^{(k)} (1 + \alpha_k)(1 - I_s)}{N_k} \le \frac{a_R^{(k)} (1 - I_s)}{f(1) N_k} = O\left(\frac{a_R^{(k)}}{N_k}\right). \tag{5.3.15}$$

Now, we use these results to analyze the convergence condition for the true channel. First, we define α_k to take the smallest value such that

$$f\left(L^{(k)}(x)\right) \lambda^{(k)}(x) < q^{(k)}(x) \tag{5.3.16}$$

for all $x \in (0, 1]$, where $L^{(k)}(x)$ and $\lambda^{(k)}(x)$ depend implicitly on α_k through the truncation of (5.3.11). Using this value for α_k means that, by definition, $\left\{\rho^{(k)}(x), \lambda^{(k)}(x)\right\}_{k \ge 0}$ is a sequence of realization degree distributions which satisfies the basic condition. Next, we derive an upper bound on the value of α_k chosen. Using (5.3.13) and (5.3.14), it is easy to verify that the condition,

$$f\left(\frac{1}{1 - \beta_k} \widetilde{L}^{(k)}(x)\right) \widetilde{\lambda}^{(k)}(x) \le q^{(k)}(x), \tag{5.3.17}$$

is more stringent than (5.3.16) and therefore implies (5.3.16). Using (5.3.8) to substitute for $q^{(k)}(x)$, we can then solve for the smallest α_k that implies (5.3.17). The result is the upper bound,

$$\alpha_k \le \max_{0 \le x \le 1} f\left(\frac{1}{1 - \beta_k} \widetilde{L}^{(k)}(x)\right) / f\left(\widetilde{L}^{(k)}(x)\right) - 1, \tag{5.3.18}$$

because any α_k which satisfies this condition must imply (5.3.17) and (5.3.16).

Finally, we show that the sequence of rate gaps $\Delta_k = I_s - R_k$ converges to zero. Using the LHS of (5.3.13) and the integral form of $a_L^{(k)}$, we can write

$$\frac{1}{a_L^{(k)}} \geq \int_0^1 \left(\widetilde{\lambda}^{(k)}(x) - \left(\widetilde{\lambda}^{(k)}(1) - 1\right) x^{N_k - 1}\right) dx = \frac{1}{\widetilde{a}_L^{(k)}} - \frac{\widetilde{\lambda}^{(k)}(1) - 1}{N_k} = (1 - \beta_k)\frac{1}{\widetilde{a}_L^{(k)}}.$$

Using this and (5.3.10), we can lower bound the code rate with

$$R_k \geq 1 - \frac{\widetilde{a}_L^{(k)}}{a_R^{(k)}(1 - \beta_k)} = 1 - \frac{(1 + \alpha_k)(1 - I_s)}{1 - \beta_k}.$$

This means that the rate gap is upper bounded by

$$\Delta_k \leq \frac{I_s(1 - \beta_k)}{1 - \beta_k} - \frac{(1 - \beta_k) - (1 + \alpha_k)(1 - I_s)}{1 - \beta_k} = \frac{(\alpha_k + \beta_k)(1 - I_s)}{1 - \beta_k}.$$

Combining the assumption that $a_R^{(k)}/N_k \to 0$ with (5.3.15) shows that $\beta_k \to 0$. Since the first derivative of $f(x)$ is bounded, we can also combine $\beta_k \to 0$ with (5.3.18) to show that $\alpha_k \to 0$. In fact, examining (5.3.18) more closely shows that $\alpha_k = O(\beta_k)$ which means the rate gap is also $\Delta_k = O(\beta_k) = O\left(a_R^{(k)}/N_k\right)$. This completes the proof. □

5.3.4 Degree Sequences with Regular Check Distributions

In this section, we limit our scope somewhat by choosing check distributions with a single non-zero coefficient. This type of check distribution is called regular, and is defined by $\rho^{(k)}(x) = x^{k-1}$. This implies that $q^{(k)}(x) = 1 - (1-x)^{1/(k-1)}$ and $Q^{(k)}(x) = (k-1)(1-x)^{k/(k-1)}/k + x$, and we use these to rewrite (5.3.11) as

$$\widetilde{\lambda}^{(k)}(x) = \frac{1 - (1-x)^{1/(k-1)}}{(1 + \alpha_k) f\left(F^{-1}\left(F(1)\frac{(k-1)(1-x)^{k/(k-1)} + kx}{k}\right)\right)}.$$

As shown in [15], the non-negative power series expansion of $1 - (1-x)^{1/(k-1)}$ is given by

$$1 - (1-x)^{1/(k-1)} = \sum_{i=1}^{\infty} \binom{1/(k-1)}{i} (-1)^{i+1} x^i.$$

This means that question of whether or not $\widetilde{\lambda}^{(k)}(x)$ has a non-negative power series expansion is very much linked to the power series expansion of $h(x) \triangleq 1/f\left(F^{-1}\left(F(1)x\right)\right)$. While answering this question is difficult, we can still make general comments. For one, this method appears to be doomed if the coefficients of $h(x)$ do not decay to zero. Since the location of the smallest

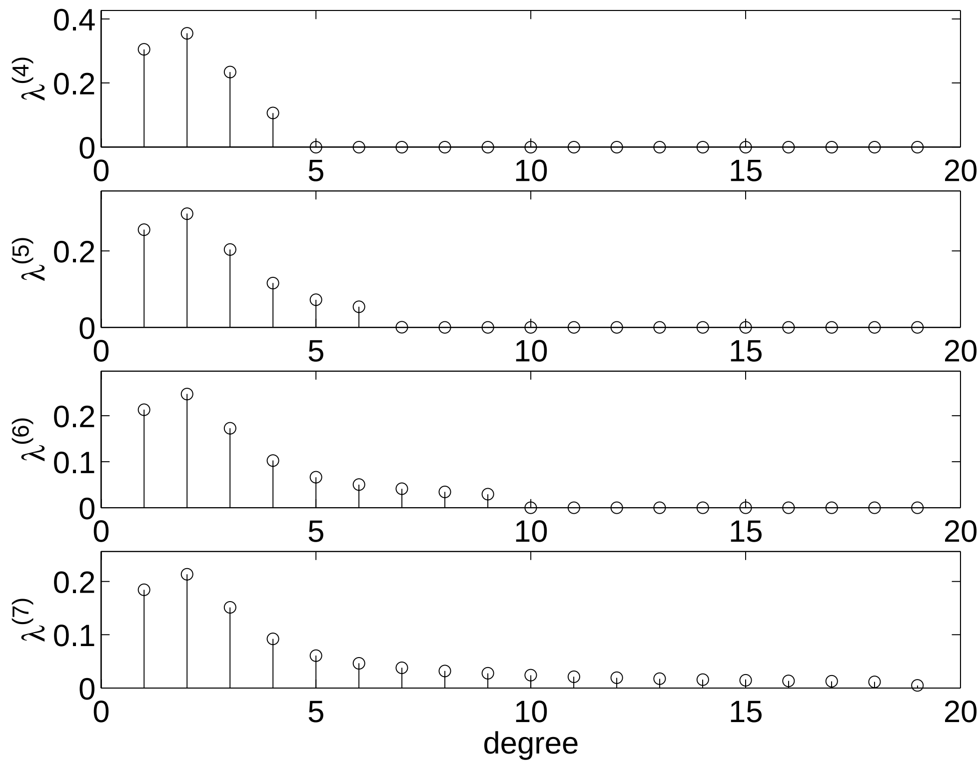

Figure 5.3.1: This shows the bit degree distributions $\lambda^{(k)}(x)$ resulting from constructing check regular codes for the precoded DEC with $\epsilon = 1/2$. The vertical axis of each subplot is scaled differently to highlight their similarity.

zero in the denominator of $h(x)$ determines the exponential growth rate of the coefficients, it makes sense that the modulus of smallest zero in the denominator should be larger than one. This implies the coefficients will decay exponentially.

For the DEC, with and without precoding, we can derive $h(x)$ in closed form. Without precoding, this expression is given by

$$h(x) = \frac{\left(x + \epsilon^2 \left(2 - 3x + 2\epsilon^3(1+x)\right)\right)^2}{\epsilon^2(1+\epsilon)^2 \left(2\epsilon^2 + x(1-\epsilon)\right)^2}.$$

The addition of a precoder changes this to

$$h(x) = \frac{(1 + 2\epsilon^2\sqrt{x(x+2-2\epsilon)} - \epsilon^2(1-2x))^2}{4\epsilon^3(x + \sqrt{x(x+2-2\epsilon)})(1 - \epsilon^2(1-x) + \epsilon^2\sqrt{x(x+2-2\epsilon)})}.$$

Using the criterion that the smallest zero of the denominator should have modulus greater than one, we have determined that the DEC without precoding requires that $0.5 < \epsilon < 1$ and the DEC

k	$a_R^{(k)}$	$a_L^{(k)}$	R_k	Δ_k	N_k	α_k	β_k	$\overline{\lambda}_1^{(k)}$	$\lambda_1^{(k)}$
4	4	1.595	0.6011	0.0655	4	0.16	0.4844	0.3127	0.3048
5	5	1.903	0.6193	0.0473	7	0.069	0.3937	0.2563	0.2562
6	6	2.102	0.6496	0.0170	9	0.048	0.2671	0.2181	0.2134
7	7	2.411	0.6555	0.0111	19	0.025	0.1681	0.1991	0.1843
8	8	2.718	0.6602	0.0064	33	0.014	0.1063	0.1615	0.1614
9	9	3.030	0.6632	0.0034	56	0.0075	0.0666	0.1436	0.1433
10	10	3.349	0.6651	0.0016	101	0.0042	0.0411	0.1287	0.1286
11	11	3.677	0.6657	0.0009	184	0.0023	0.0249	0.1166	0.1165

Table 5.1: Code construction results for the precoded DEC with $\epsilon = 1/2$.

with precoding requires that $0 < \epsilon \lesssim 0.6309$. While this does not prove that sequences of check regular codes cannot achieve the SIR of arbitrary GECs, it definitely hints at this possibility.

For the precoded DEC with $\epsilon = 0.5$, we constructed the check regular code sequence for $k = 4, \ldots, 11$. While we were unable to prove that the coefficients of each power series expansion are non-negative, we did verify this numerically for the first 200 coefficients. The results of this experiment are shown in Table 5.1 and Figure 5.3.1. Since the choice of α_k in our construction guarantees that each code satisfies the convergence condition for the channel, all of these rates and rate gaps are valid. It is also worth noting that the degree sequences shown in Figure 5.3.1 are all very similar once you account for truncation and scaling. The table also gives the fraction of edges connected to degree 1 nodes, $\lambda_1^{(k)}$, and the maximum value allowed by the stability condition, $\overline{\lambda}_1^{(k)}$. Although, we cannot prove that this sequence of codes satisfies the conditions of Theorem 5.3.4, we can still compare the results to the predictions of the theorem. We find that the constants follow the predictions of Theorem 5.3.4 quite well. For one, N_k appears to be growing exponentially with k and Δ_k seems to be decaying exponentially with k. This type of behavior is well-known for check regular codes on the BEC [16].

For the DEC with $\epsilon = 0.85$ and no precoding, we also constructed the check regular code sequence. In this case, N_k grows so rapidly that we could only construct the codes with $k = 3, 4, 5$. This time, we verified numerically that the first 800 coefficients of each power series are non-negative. The results are shown in Table 5.2, and again the sequences follow the predictions of Theorem 5.3.4 quite well. This table also shows the fraction of edges connected

k	$a_R^{(k)}$	$a_L^{(k)}$	R_k	Δ_k	N_k	α_k	β_k	$\overline{\lambda}_2^{(k)}$	$\lambda_2^{(k)}$
3	3	2.370	0.2101	0.0088	14	0.00053	0.0308	0.6920	0.6916
4	4	3.129	0.2177	0.0011	107	0.00018	0.0054	0.4613	0.4612
5	5	3.906	0.2187	0.0002	757	0.00003	0.0009	0.3460	0.3460

Table 5.2: Code construction results for the DEC with $\epsilon = 0.85$ and no precoding.

to degree 2 bit nodes, $\lambda_2^{(k)}$, and the maximum value allowed by the stability condition, $\overline{\lambda}_2^{(k)}$.

Finally, we should note that values of ϵ and k chosen for these experiments carefully avoided power series expansions with non-negative coefficients. Though not shown here, we have also considered code sequences with arbitrary ETFs such as $f(x) = ax + b$. In all these cases, we have not found any situation where the power series expansion suddenly has a negative term after a long initial sequence of positive terms. In general, the expansions we have found with negative terms expose themselves within the first five terms. Picking larger values of k also seems to help and any expansion usually has negative terms for small enough k.

5.4 Results for Arbitrary GECs

5.4.1 The Existence of Arbitrary GECs

From what we have discussed so far, it is not clear that the set of GECs contains anything more than the DEC with and without precoding. Nothing in our analysis, however, prevents us from considering the much larger family of GECs implied by any non-decreasing $f(x)$ which maps the interval $[0, 1]$ into itself. Moreover, we believe that it is possible to construct, albeit somewhat artificially, a binary-input GEC for any such $f(x)$. This would means that, in some sense, there is a GEC for every well-defined ETF. Similar ideas may also be useful in the context of EXIT charts analysis.

The ETF of a GEC is defined as mapping from the *a priori* erasure probability of the channel decoder to the erasure probability of the extrinsic output. If the *a priori* messages from the code to the channel decoder are divided randomly into two groups of equal size, then erasure probability in the two groups will be the same. Now, suppose that these groups of bits are sent through different GECs. In this case, the extrinsic messages from the first channel will have

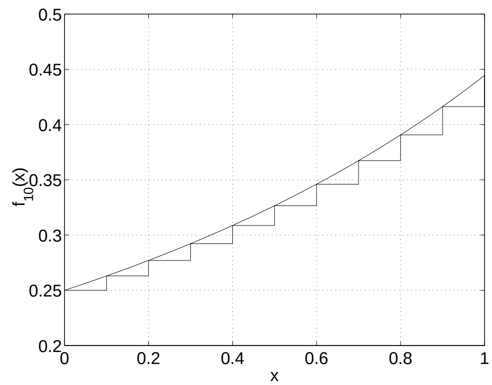

Figure 5.4.1: The results of approximating the non-precoded DEC with $\epsilon = 0.5$ and $n = 10$.

erasure probability $f_1(x)$ and the extrinsic messages from the second channel will have erasure probability $f_2(x)$. Since the two groups were chosen at random, the average erasure probability of all the extrinsic messages passed back to the code will be $(f_1(x) + f_2(x))/2$. This idea of linearly combining channels was first introduced in the context of EXIT charts and doping [17]. It also extends naturally to an arbitrary weighted combinations of GECs.

Now, consider the performance of a rate-1/2 systematic linear code when the systematic bits are erased with probability x and the non-systematic bits are erased with probability y. We assume that the code is chosen randomly and that the block length is arbitrarily large. Assuming that the coding theorem holds for this special case, a maximum likelihood decoder should be able to recover all of the systematic bits as long as the average bit erasure rate, $(x + y)/2$, is less than $1/2$. If the average erasure rate is larger than $1/2$, then, with probability 1, there will not be enough information to reconstruct all of the systematic bits. Therefore, we conjecture that the extrinsic erasure rate, at the output of an APP decoder for this code, will be given by

$$u(x, y) = \begin{cases} 0 & 0 \le x + y < 1 \\ 1 & 1 \le x + y \le 2 \end{cases} .$$

This ensemble of codes can be treated as a GEC whose inputs are the systematic bits and whose outputs are the parity bits. In this case, the ETF for the GEC with parameter y and *a priori* erasure rate x is given by $u(x, y)$. It is not too hard to see that we can approximate any non-decreasing $f(x)$, which maps the interval $[0, 1]$ into itself, in this fashion. If we define the sequence of approximations by

$$f_n(x) = \sum_{i=0}^{n} w_n\left(\frac{i}{n}\right) u\left(x, \frac{i}{n}\right)$$

with

$$w_n\left(\frac{i}{n}\right) = f\left(\frac{n-i}{n}\right) - f\left(\frac{n-i-1}{n}\right),$$

then $f_n(x)$ is essentially the nth order lower Riemann approximation of $f(x)$. An example of the approximating function is shown in Figure 5.4.1. If $f(x)$ is differentiable on $[0, 1]$, these approximations will converge, as n goes infinity, to

$$f(x) = \int_0^1 w(y)u(x, y)dy$$

with

$$w(y) = \delta(1-y)f(0) + f'(1-y),$$

where $\delta(y)$ is the Dirac delta function.

5.4.2 Numerical Optimization via Linear Programming

In this section, we attempt to leverage our closed form condition, (5.3.6), into a practical technique for optimizing degree distributions for arbitrary GECs. The result is an algorithm similar to the alternating LP optimization presented in [11]. The optimization algorithm works by choosing a regularly spaced grid of points and using LP to maximize the code rate while satisfying the constraint at each grid point. We use a grid based on the erasure probability passed into a check node, make use of the fact that (5.3.2) need only be satisfied for $x \in (0, f(1)]$. Using this, it is sufficient to use the grid $\mathcal{X} = \{0, s, 2s, \ldots, \lfloor f(1)/s \rfloor s, f(1)\}$, and we note that $s = 0.02$ seems to work well in practice. The arguments passed to the algorithm consist of the set of active bit degrees $\mathcal{L}$, the set of active check degrees $\mathcal{R}$, and the initial check degree distribution $\rho(x)$. The algorithm proceeds by alternately optimizing $\lambda(x)$, for fixed $\rho(x)$, and $\rho(x)$, for fixed $\lambda(x)$.

There are three things which make this algorithm tricky to use in practice, however. The first is that the parameter a_L in (5.3.7) must be guessed correctly to make things work. The second is that the function $q(x)$ generally gets very steep as x approaches 1, so expressions must be evaluated carefully to avoid numerical problems. The third is that LP is used to maximize the rate under a constraint similar to (5.3.7), but we know this constraint does not imply that (5.3.2) holds as well. The algorithm is stabilized by the fact that the $\rho(x)$ optimization always produces a valid code which satisfies (5.3.2). Furthermore, if the algorithm converges, then (5.3.7) will be satisfied with near equality which implies that (5.3.2) will also be satisfied with near equality. All of these add up to an algorithm that can work quite well, albeit with some tweaking of the initial conditions. Lastly, the convergence is rarely monotonic and the algorithm may initially wander through terrible codes (i.e. negative rate) before finding a very good code.

Consider optimizing $\rho(x)$ for some fixed $\lambda(x)$. The goal of maximizing the code rate for fixed $\lambda(x)$ is equivalent to minimizing the linear objective function,

$$1/a_R = \sum_{\nu \in \mathcal{R}} \rho_v \frac{1}{\nu}, \tag{5.4.1}$$

because the code rate is $1 - a_L/a_R$. The LP constraints can be derived by starting with (5.3.4) and applying $q^{-1}(x)$ to both sides to get

$$q^{-1}\left(f\left(L(x)\right)\lambda(x)\right) < x.$$

Since $q^{-1}(x) = 1 - \rho(1-x)$, this inequality can be rewritten as

$$\sum_{\nu \in \mathcal{R}} \rho_\nu \left(1 - \left(1 - f\left(L(x)\right)\lambda(x)\right)^{\nu-1}\right) < x,$$

which is linear in the ρ_ν's. In practice, we have found that the numerical robustness is improved by letting $\phi(x)$ equal the $\rho(x)$ from the previous iteration, and requiring that

$$\sum_{\nu \in \mathcal{R}} \rho_\nu \left(1 - \left(1 - f\left(L\left(1-\phi(1-x)\right)\right)\lambda\left(1-\phi(1-x)\right)\right)^{\nu-1}\right) < 1 - \phi(1-x) \tag{5.4.2}$$

be satisfied for all $x \in \mathcal{X}$. The stability condition, (5.3.3), can also be handled via the extra inequality

$$\sum_{\nu \in \mathcal{R}} \rho_\nu (\nu - 1) < \begin{cases} 1/\left(\lambda_1^2 f'(0) a_L\right) & \text{if } f(0) = 0 \\ 1/\left(\lambda_2 f(0)\right) & \text{if } f(0) > 0 \end{cases}.$$

Now, we consider optimizing $\lambda(x)$ for fixed $\rho(x)$. Our goal of maximizing the code rate for fixed $\rho(x)$ is equivalent to maximizing the linear objective function,

$$1/a_L = \sum_{\nu \in \mathcal{L}} \lambda_v \frac{1}{\nu}, \tag{5.4.3}$$

because the code rate is $1 - a_L/a_R$. The LP constraints can be derived by starting with (5.3.7), substituting $1-\rho(1-x)$ for x, and transforming it into an inequality to get

$$\lambda\left(1-\rho(1-x)\right) < \frac{x}{f\left(F^{-1}\left(a_L Q\left(1-\rho(1-x)\right)\right)\right)}.$$

One subtlety is choosing the a_L so the algorithm performs well. We have found that choosing $a_L = F(1)/a_R$, which implicitly assumes that we will achieve the SIR, does the trick. Adding the relaxation constant, c, and rewriting this gives the linear inequality,

$$\sum_{\nu\in\mathcal{L}} \lambda_\nu \left(1-\rho(1-x)\right)^{\nu-1} < \frac{cx}{f\left(F^{-1}\left(F(1)a_R Q\left(1-\rho(1-x)\right)\right)\right)}, \tag{5.4.4}$$

which must be satisfied for all $x \in \mathcal{X}$. It is worth pointing out the similarity between (5.4.4) and (5.3.11) with c playing the role of α_k. The stability condition, (5.3.3), depends on the channel and is given by

$$\begin{array}{ll} \lambda_1 < c\sqrt{\frac{1}{f'(0)a_L\rho'(1)}} & \text{if } f(0)=0 \\ \lambda_2 < \frac{c}{f(0)\rho'(1)} & \text{if } f(0)>0 \end{array}$$

The relaxation constant is used to improve convergence and we typically use $c = 1-(0.1)^{1+i/4}$ for the ith $\lambda(x)$ optimization.

In our implementation, all of the function evaluations required are done via linear interpolation from sampled function tables. For example, we start by sampling $f(x)$ on a very fine grid. Next, we let $F(x)$ be the integral of linear interpolated $f(x)$, which is given by trapezoidal integration of the $f(x)$ sample table. Inverse functions can also be handled by via linear interpolation. To do this, one simply reverses the role of the sampling grid and the sampled function table. Finally, the function $Q\left(1-\rho(1-x)\right)$ can be computed accurately and efficiently using

$$Q\left(1-\rho(1-x)\right) = \sum_{\nu\in\mathcal{R}} \rho_\nu \left(\frac{1-(1-x)^\nu}{\nu} - x(1-x)^{\nu-1}\right). \tag{5.4.5}$$

Remark 5.4.1. This formula for $Q\left(1-\rho(1-x)\right)$ can be derived by noticing that the rectangle extending from $(0,0)$ to $(x, q(x))$ can be divided into two regions by the curve $q(t)$ with $t \in [0,x]$. Using the fact that the area of these two regions must sum to $xq(x)$, we have

$$\int_0^x q(t)dt + \int_0^{q(x)} q^{-1}(t)dt = xq(x).$$

This allows us to compute the integral of $q(x)$ with

$$\begin{aligned} Q(x) &= \int_0^x q(t)dt \\ &= xq(x) - \int_0^{q(x)} q^{-1}(t)dt \\ &= xq(x) - \int_0^{q(x)} \left(1-\rho(1-t)\right)dt \\ &= xq(x) - \sum_{\nu\geq 1} \rho_\nu \left(q(x) - \frac{1-(1-q(x))^\nu}{\nu} \right). \end{aligned}$$

Finally, we evaluate this expression by substituting $1-\rho(1-y) = q^{-1}(y)$ for x to get

$$Q\left(1-\rho(1-y)\right) = \left(1-\rho(1-y)\right) y - \sum_{\nu\geq 1} \rho_\nu \left(y - \frac{1-(1-y)^\nu}{\nu} \right).$$

Expanding $\rho(1-y)$ and simplifying gives (5.4.5).

5.4.3 A Stability Condition for General Channels

In this section, we discuss one implication that this research has on the joint iterative decoding of LDPC codes and general channels with memory. This implication is that the stability condition for general channels may actually be as simple as the stability condition for memoryless channel. Recall that, as long as $f(0) > 0$, the stability condition for GECs is given by $\lambda_2 \rho'(1) f(0) < 1$. This condition is identical to the stability condition for the memoryless erasure channel with erasure probability $f(0)$. Let $F_0(x)$ be the LLR density at the extrinsic output of the channel decoder, for a general channel, when perfect *a priori* information is passed to the decoder. As long as $F_0(x)$ does not have perfect information itself (i.e., it is not equal to a delta function at infinity), then the stability condition is given by applying the memoryless channel condition from [13] to $F_0(x)$. This makes sense because, when the joint decoder is near convergence, the LLRs passed as *a priori* information to the channel decoder are nearly error free. A more rigorous analysis of this phenomenon is given in Appendix 5B.3.

5.5 Concluding Remarks

In this chapter, we consider the joint iterative decoding of irregular LDPC codes and channels with memory. We introduce a new class of erasure channels with memory, known as generalized erasure channels (GECs). For these channels, we derive a single parameter recursion for density evolution of the joint iterative decoder. This allows us to state necessary and sufficient conditions for decoder convergence and to algebraically construct sequences of LDPC degree distributions which appear to approach the symmetric information rate of the channel. This provides the first possibility of proving that the SIR is actually achievable via iterative decoding. In the future, we hope to prove that the two degree sequences constructed in this chapter actually achieve the SIR. The bigger question is whether or not it is possible to construct degree distribution sequences which achieve the SIR for any GEC.

5A Exact Analysis of the BCJR Decoder for the DEC

In this section, we analyze the behavior of a BCJR decoder for a DEC with erasure rate ϵ. This is achieved by first finding the steady state distributions of the forward and backward recursions, and then computing the extrinsic erasure rate of the decoded input stream. Prior knowledge of the input is taken into account by assuming that it is observed independently through a BEC with erasure rate δ. This approach makes it possible to derive closed form expressions for a system which combines a low density parity check (LDPC) code with a BCJR decoder for this channel. Throughout this section, we assume that the channel inputs are chosen i.i.d. $B(1/2)$.

5A.1 The Dicode Erasure Channel without Precoding

Consider the BCJR algorithm operating on a DEC without precoding. In this section, we compute the extrinsic erasure rate of that decoder as an explicit function of the channel erasure rate, ϵ, and the *a priori* erasure rate, δ. This is done by analyzing the forward recursion, the backward recursion, and the output stage separately.

Expanding our decoder to consider *a priori* information is very similar to expanding the alphabet of our channel. Instead of receiving a single output symbol from the set $\mathbb{Y} = \{-, 0, +, e\}$, we receive a pair of output symbols. One is from the set $\mathbb{Y}$ and the other, which represents the *a priori* symbol, is from the set $\mathbb{W} = \{0, 1, e\}$. Since the channel has only two states, it suffices to consider the quantity $\alpha^{(t)} \triangleq \alpha_0^{(t)} = 1 - \alpha_1^{(t)} = Pr(S_t = 0|\mathbf{W}_1^{t-1}, \mathbf{Y}_1^{t-1})$. The real simplification, however, comes from the fact that the distribution of $\alpha^{(t)}$ has finite support when $X \sim B(1/2)$. Let W_t and Y_t be the *a priori* symbol and channel output received at time t, respectively. Using this, we can write the forward recursion as

$$\boldsymbol{\alpha}^{(t+1)} = \frac{\boldsymbol{\alpha}^{(t)}\mathbf{M}_\alpha(W_t, Y_t)}{\left\|\boldsymbol{\alpha}^{(t)}\mathbf{M}_\alpha(W_t, Y_t)\right\|_1},$$

where $\boldsymbol{\alpha}^{(t)} = \left[\alpha^{(t)} \;\; 1-\alpha^{(t)}\right]$ and $\left[\mathbf{M}_\alpha(w,y)\right]_{ij} = Pr(S_{t+1}=j, W_t=w, Y_t=y|S_t=i)$. It is easy to verify that this recursion is identical to the simpler recursion,

$$\alpha^{(t+1)} = \begin{cases} 1/2 & \text{if } Y_t = e \text{ and } W_t = e \\ \alpha^{(t)} & \text{if } Y_t = 0 \text{ and } W_t = e \\ 0 & \text{if } Y_t = + \text{ or } W_t = 1 \\ 1 & \text{if } Y_t = - \text{ or } W_t = 0 \end{cases}.$$

Using the simple recursion, we see that, for all $t \geq \min\{i \geq 1 | Y_i \neq 0 \text{ or } W_i \neq e\}$, $\alpha^{(t)}$ will be confined to the finite set $\{0, 1/2, 1\}$.

The inherent symmetry of the channel actually allows us to consider even a smaller support set. The real difference between the three α values in the support set is whether the state is known perfectly or not. When $\alpha^{(t)} \in \{0,1\}$, the state is known with absolute confidence, while $\alpha^{(t)} = 1/2$ corresponds to no prior knowledge.

Using a two state Markov chain, we can compute the steady state probabilities of a new Markov chain which characterizes the forward recursion. To do this, we treat the known state condition (i.e., $\alpha^{(t)} \in \{0,1\}$) as the K_α state and unknown state condition (i.e., $\alpha^{(t)} = 1/2$) as the U_α state. The new Markov chain transitions from the K_α state to the U_α state only if $W = e$ and $Y = e$. Therefore, we have $Pr(K_\alpha \to U_\alpha) = 1 - Pr(K_\alpha \to K_\alpha) = \epsilon\delta$. The new Markov chain also transitions from the U_α state to the U_α state only if $W = e$ and $Y \in \{e, 0\}$. This means that we have $Pr(U_\alpha \to U_\alpha) = 1 - Pr(U_\alpha \to K_\alpha) = \delta(\epsilon + (1-\epsilon)/2)$. The steady state probabilities $Pr(K_\alpha)$ and $Pr(U_\alpha)$ can be found using the eigenvector equation,

$$\left[\begin{array}{cc} Pr(K_\alpha) & Pr(U_\alpha) \end{array}\right] \left[\begin{array}{cc} 1-\epsilon\delta & \epsilon\delta \\ 1-\frac{\delta(1+\epsilon)}{2} & \frac{\delta(1+\epsilon)}{2} \end{array}\right] = \left[\begin{array}{cc} Pr(K_\alpha) & Pr(U_\alpha) \end{array}\right],$$

whose solution is $Pr(U_\alpha) = 1 - Pr(K_\alpha) = \frac{2\epsilon\delta}{2-\delta(1+\epsilon)+2\epsilon\delta}$.

The backward recursion is analyzed in an almost identical manner. In this case, it suffices to consider the quantity $\beta^{(t)} \triangleq \beta_0^{(t)} = 1 - \beta_1^{(t)} = Pr(S_t = 0|\mathbf{W}_t^n, \mathbf{Y}_t^n)$. Now, we can write the backward recursion as

$$\boldsymbol{\beta}^{(t+1)} = \frac{\boldsymbol{\beta}^{(t)}\mathbf{M}_\beta(W_t, Y_t)}{\left\|\boldsymbol{\beta}^{(t)}\mathbf{M}_\beta(W_t, Y_t)\right\|_1},$$

where $\boldsymbol{\beta}^{(t)} = \left[\beta^{(t)} \;\; 1-\beta^{(t)}\right]$ and $\left[\mathbf{M}_\beta(w,y)\right]_{ij} = Pr(S_t = j, W_t = w, Y_t = y | S_{t+1} = i)$. Again, we have a simpler recursion which, in this case, is given by

$$\beta^{(t+1)} = \begin{cases} 1/2 & \text{if } Y_t = e \\ \alpha^{(t)} & \text{if } Y_t = 0 \text{ and } W_t = e \\ 0 & \text{if } Y_t = + \text{ or } (Y_t = 0 \text{ and } W_t = 0) \\ 1 & \text{if } Y_t = - \text{ or } (Y_t = 0 \text{ and } W_t = 1) \end{cases}.$$

Using the simple recursion, we see that, for all $t \geq \min\{i \geq 1 | Y_i \neq 0\}$, $\beta^{(t)}$ will be confined to the finite set $\{0, 1/2, 1\}$.

Now, we can use a two state Markov chain to compute the steady state probabilities of a new Markov chain which characterizes the backward recursion. To do this, we treat the known state condition (i.e., $\beta^{(t)} \in \{0,1\}$) as the K_β state and unknown state condition (i.e., $\beta^{(t)} = 1/2$) as the U_β state. The new Markov chain transitions from the K_β state to the U_β state if $Y = e$. Therefore, we have $Pr(K_\beta \to U_\beta) = 1 - Pr(K_\beta \to K_\beta) = \epsilon$. The new Markov chain also transitions from the U_β state to the U_β state if: (i) $Y = e$ or (ii) $W = e$ and $Y = 0$. This means that we have $Pr(U_\beta \to U_\beta) = 1 - Pr(U_\beta \to K_\beta) = \epsilon + \delta(1-\epsilon)/2$. The steady state probabilities $Pr(K_\alpha)$ and $Pr(U_\alpha)$ can be found using the eigenvector equation,

$$\left[\begin{array}{cc} Pr(K_\beta) & Pr(U_\beta) \end{array}\right] \left[\begin{array}{cc} 1-\epsilon & \epsilon \\ 1-\epsilon-\frac{\delta(1-\epsilon)}{2} & \epsilon+\frac{\delta(1-\epsilon)}{2} \end{array}\right] = \left[\begin{array}{cc} Pr(K_\beta) & Pr(U_\beta) \end{array}\right],$$

whose solution is $Pr(U_\beta) = 1 - Pr(K_\beta) = \frac{2\epsilon}{(1-\epsilon)(2-\delta)+2\epsilon}$.

Now, we consider the output stage of the BCJR algorithm DEC without precoding. At any point in the trellis, there are now four distinct possibilities for forward/backward state knowledge: $K_\alpha K_\beta$, $K_\alpha U_\beta$, $U_\alpha K_\beta$, and $U_\alpha U_\beta$. At the extrinsic output of the decoder, the respective erasure probability conditioned on each possibility is: 0, ϵ, 0, and $(1+\epsilon)/2$. Therefore, the erasure probability of the extrinsic output of the BCJR is

$$\begin{aligned} P_e &= Pr(U_\beta)\left(\epsilon Pr(K_\alpha) + \frac{1+\epsilon}{2}Pr(U_\alpha)\right) \\ &= \frac{2\epsilon}{(1-\epsilon)(2-\delta)+2\epsilon}\left(\epsilon\frac{2-\delta(1+\epsilon)}{2-\delta(1+\epsilon)+2\epsilon\delta} + \frac{1+\epsilon}{2}\frac{2\epsilon\delta}{2-\delta(1+\epsilon)+2\epsilon\delta}\right) \\ &= \frac{4\epsilon^2}{(2-\delta(1-\epsilon))^2}. \end{aligned}$$

Decoding without *a priori* information is equivalent to choosing $\delta = 1$, and the corresponding expression simplifies to $4\epsilon^2/(1+\epsilon)^2$.

5A.2 The Dicode Erasure Channel with Precoding

Consider the BCJR algorithm operating on a DEC using a $1/(1+D)$ precoder. In this section, we compute the extrinsic erasure rate of that decoder as an explicit function of the channel erasure rate, ϵ, and the *a priori* erasure rate, δ. This is done by analyzing the forward recursion, the backward recursion, and the output stage separately.

Our approach is basically the same as that of Section 5A.1. Therefore, we start by simplifying things with the reduced forward recursion given by

$$\alpha^{(t+1)} = \begin{cases} 1/2 & \text{if } Y_t = e \text{ and } W_t = e \\ \alpha^{(t)} & \text{if } Y_t = 0 \text{ or } Z_t = 0 \text{ or } Z_t = 1 \\ 0 & \text{if } Y_t = + \\ 1 & \text{if } Y_t = - \end{cases}.$$

Using this, we see that, for all $t \geq \min\{i \geq 1 | Y_i \neq 0\}$, $\alpha^{(t)}$ will be confined to the finite set $\{0, 1/2, 1\}$.

The two state Markov chain that we use to characterize the forward recursion is again based on the known state condition K_α (i.e., $\alpha^{(t)} \in \{0,1\}$) and the unknown state condition U_α (i.e., $\alpha^{(t)} = 1/2$). In this case, the new Markov chain transitions from the K_α state to the U_α state only if $W = e$ and $Y = e$. Therefore, we have $Pr(K_\alpha \to U_\alpha) = 1 - Pr(K_\alpha \to K_\alpha) = \epsilon\delta$. The new Markov chain also transitions from the U_α state to the K_α state only if $Y \in \{+,-\}$. This means that we have $Pr(U_\alpha \to K_\alpha) = 1 - Pr(U_\alpha \to U_\alpha) = (1-\epsilon)/2$. The steady state probabilities $Pr(K_\alpha)$ and $Pr(U_\alpha)$ can be found using the eigenvector equation,

$$\begin{bmatrix} Pr(K_\alpha) & Pr(U_\alpha) \end{bmatrix} \begin{bmatrix} 1-\epsilon\delta & \epsilon\delta \\ \frac{(1-\epsilon)}{2} & \frac{(1+\epsilon)}{2} \end{bmatrix} = \begin{bmatrix} Pr(K_\alpha) & Pr(U_\alpha) \end{bmatrix},$$

whose solution is $Pr(K_\alpha) = 1 - Pr(U_\alpha) = \frac{1-\epsilon}{1-\epsilon+2\epsilon\delta}$.

The precoded case is also simplified by the fact that the state diagram of the precoded channel is such that time reversal is equivalent to negating the sign of the output. Therefore, the statistics of the forward and backward recursions are identical and $Pr(K_\beta) = 1 - Pr(U_\beta) = Pr(K_\alpha)$.

Now, we consider the output stage of the BCJR algorithm for the precoded DEC. At any point in the trellis, there are now four distinct possibilities for forward/backward state knowledge: $K_\alpha K_\beta$, $K_\alpha U_\beta$, $U_\alpha K_\beta$, and $U_\alpha U_\beta$. At the extrinsic output of the decoder, the respective erasure probability conditioned on each possibility is: 0, ϵ, ϵ, and ϵ. Therefore, the erasure probability of the extrinsic output of the BCJR is

$$\begin{aligned} P_e &= \epsilon\left(1 - Pr(K_\alpha)Pr(K_\beta)\right) \\ &= \epsilon\left(1 - \frac{(1-\epsilon)^2}{(1-\epsilon+2\epsilon\delta)^2}\right) \\ &= \frac{4\epsilon^2\delta(1-\epsilon(1-\delta))}{(1-\epsilon(1-2\delta))^2}. \end{aligned}$$

Again, decoding without *a priori* information is equivalent to choosing $\delta = 1$, and the corresponding expression simplifies to $4\epsilon^2/(1+\epsilon)^2$.

5B Joint Iterative Decoding DE for General Channels

While the initial steps of this analysis were focused on GECs, it is entirely possible to write out the DE recursion, such as (5.3.1), for general message passing. Many of these ideas were first introduced by Richardson and Urbanke in [13]. This recursion will track the density function of the messages passed along a particular edge. In general, the messages will be LLRs but most of this analysis does not depend on this. We will, however, make use of notation based on using LLR messages. For example, we use Δ_∞ to denote the message implying perfect knowledge of a "0" bit because, in the LLR domain, this message corresponds to a delta function at infinity. Likewise, the message implying perfect knowledge of a "1" bit is denoted by $\Delta_{-\infty}$. We also use Δ_0 to denote the message implying the complete lack of knowledge because, in the LLR domain, this message corresponds to a delta function at zero.

For channels with memory, the standard DE assumption of channel symmetry may not hold. Essentially, this means that DE can only be applied to one codeword at a time. In [7], the i.i.d. channel adaptor is introduced as a conceptual device which ensures the symmetry of any channel. If the outer code is a linear code, then this approach is identical to choosing a random coset and treating it as part of the channel. In this section, we use the i.i.d. channel adaptor approach so that DE can be applied to all codewords simultaneously.

Let $\otimes$ be a commutative binary operator on message densities which represents the message combining function for the bit nodes. For example, if the messages consist of LLRs, then this operator will represent convolution because LLRs are added at the bit nodes. In this case, the operator can be defined by noting that $R = P \otimes Q$ implies

$$R(x) = \int_{-\infty}^{\infty} P(y)Q(x-y)dy.$$

Using the notation,

$$P^{\otimes k} = \underbrace{P \otimes \ldots \otimes P}_{k \text{ times}},$$

for exponentials, we can now define $\lambda^{\otimes}(P) = \sum_{\nu \geq 1} \lambda_\nu P^{\otimes \nu - 1}$ and $L^{\otimes}(P) = \sum_{\nu \geq 1} L_\nu P^{\otimes \nu}$. Let $\oplus$ be a commutative binary operator on message densities which represents the message combining function for the check nodes. If min-sum decoding is used in the LLR domain, then this operator can be defined by noting that $R = P \oplus Q$ implies $R(x) = R_+(x)U(x) + R_-(x)U(-x) + R_0(x)\Delta_0(x)$, where

$$\begin{aligned} R_+(x) &= P(x)\int_x^{\infty} Q(y)dy + P(-x)\int_{-\infty}^{-x} Q(-y)dy \\ R_0 &= \int_{0^-}^{0^+} P(y)dy + \int_{0^-}^{0^+} Q(y)dy - \int_{0^-}^{0^+} P(y)Q(y)dy \\ R_-(x) &= P(x)\int_{-\infty}^{x} Q(y)dy + P(-x)\int_x^{\infty} Q(y)dy, \end{aligned}$$

and $U(x)$ is the unit step function. Using similar notation for operator exponentials, we also define $\rho^{\oplus}(P) = \sum_{\nu \geq 1} \rho_\nu P^{\oplus \nu - 1}$. We also require that both of these operators be commutative and associative with respect to scalar multiplication. This means that

$$cP \otimes Q = P \otimes cQ = c(P \otimes Q),$$

and that the respective result holds for $\oplus$.This allows us to apply the natural analogue of the binomial theorem to show that

$$(aP \oplus bQ)^k = \sum_{i=0}^{k} \binom{k}{i} a^i b^{k-i} P^{\oplus i} \oplus P^{\oplus k-i}, \tag{5B.1}$$

and that the respective result holds for $\otimes$. We also note that the zero power gives the identity, so $P^{\otimes 0} = \Delta_0$ and $P^{\oplus 0} = \Delta_\infty$.

Our next assumption, known as *decoder symmetry*, requires that the operators for the bit and check nodes obey a few identities. The bit node operator must satisfy $P \otimes \Delta_0 = P$, $P \otimes \Delta_\infty = \Delta_\infty$, and $P \otimes \Delta_{-\infty} = \Delta_{-\infty}$. This means, in some sense, that Δ_0 acts as an identity and that Δ_∞ (and $\Delta_{-\infty}$) act as zero elements. The check node operator must satisfy $P \otimes \Delta_\infty = P$, $P \otimes \Delta_0 = \Delta_0$, and the fact that $P = Q \otimes \Delta_{-\infty}$ implies $P(x) = Q(-x)$. This means, in some sense, that Δ_∞ acts as the identity, $\Delta_{-\infty}$ acts as a negative identity, and Δ_0 acts as the zero element. We note that most reasonable combining functions for the bit and check nodes satisfy these properties.

Finally, we let $\mathfrak{F}(P)$ represent the mapping from *a priori* messages to extrinsic messages for the channel decoder. Using the same arguments used for (5.3.1), we can now write the DE recursion as

$$P_{i+1} = \mathfrak{F}\left(L^{\otimes}\left(\rho^{\oplus}(P_i)\right)\right) \otimes \lambda^{\otimes}\left(\rho^{\oplus}(P_i)\right). \tag{5B.2}$$

A sufficient condition for convergence can be defined by requiring, for example, that the hard decision error probability is decreased by every iteration. The hard decision error probability for a density is simply a projection of that density onto a scalar, and there are a variety of such projections which can be used to define sufficient conditions for convergence. In the LLR domain, entropy of the bit given the message is another projection that seems to work well.

Using the i.i.d. channel adaptor approach ensures that the expected value of the decoder trajectory does not depend on the codeword transmitted. Since the choice of the random coset is absorbed into the channel, the all zero bit pattern still acts as a codeword and a fixed point of the iteration. Therefore, the message density Δ_∞ is always a fixed point of the iteration and we can discuss its stability. In the same manner as LDPC codes for memoryless channels [13], we can expand this recursion in a series about Δ_∞ by letting $P_i = (1-\epsilon)\Delta_\infty + \epsilon Q$. Our only new assumption is that the channel mapping can be expanded about this density with

$$\mathfrak{F}\left((1-\epsilon)\Delta_\infty + \epsilon Q\right) = (1 - c_F(Q)\epsilon)F_0 + c_F(Q)\epsilon D_F(Q) + O(\epsilon^2),$$

where F_0 is the perfect a priori channel LLR density, $c_F(Q)$ is a scalar valued function of a density, and $D_F(Q)$ is a density valued function of a density. We note that F_0 can be estimated in practice by simulating the channel decoder with perfect *a priori* information. The functions

$c_F(Q)$ and $D_F(Q)$ will usually depend in a complicated way on Q, but we can simplify things by defining $c_F = \sup_Q c_F(Q)$. Using simulations, more accurate results may be possible by estimating $c_F(Q)$ and $D_F(Q)$ along a particular decoding trajectory.

Now, we can use (5B.1) to expand the operators $\rho^{\oplus}$, $\lambda^{\otimes}$, and $L^{\otimes}$ around the density $(1-\epsilon)\Delta_\infty + \epsilon Q$. For $\rho^{\oplus}$, this gives

$$\begin{aligned}\rho^{\oplus}\left((1-\epsilon)\Delta_\infty + \epsilon Q\right) &= \sum_{\nu \geq 1} \rho_\nu \sum_{i=0}^{\nu-1} \binom{\nu-1}{i} (1-\epsilon)^{\nu-1-i} \epsilon^i \Delta_\infty^{\oplus \nu-1-i} \oplus Q^{\otimes i} \\ &= \sum_{\nu\geq 1} \rho_\nu (1-\epsilon)^{\nu-1}\Delta_\infty + \rho'(1)\epsilon Q + O(\epsilon^2),\end{aligned}$$

because $\sum_{\nu\geq 1} \rho_\nu(\nu-1) = \rho'(1)$. For $\lambda^{\otimes}$, this gives

$$\begin{aligned}\lambda^{\otimes}\left((1-\epsilon)\Delta_\infty + \epsilon Q\right) &= \sum_{\nu \geq 1} \lambda_\nu \sum_{i=0}^{\nu-1} \binom{\nu-1}{i} (1-\epsilon)^{\nu-1-i} \epsilon^i \Delta_\infty^{\otimes \nu-1-i} \otimes Q^{\otimes i} \\ &= \lambda_1 \Delta_0 + \sum_{\nu\geq 2} \lambda_\nu (1-\epsilon^{\nu-1})\Delta_\infty + \lambda_2 \epsilon Q + O(\epsilon^2).\end{aligned}$$

For $L^{\otimes}$, this gives

$$\begin{aligned}L^{\otimes}\left((1-\epsilon)\Delta_\infty + \epsilon Q\right) &= \sum_{\nu \geq 1} L_\nu \sum_{i=0}^{\nu} \binom{\nu}{i} (1-\epsilon)^{\nu-i} \epsilon^i \Delta_\infty^{\otimes \nu-i} \otimes Q^{\otimes i} \\ &= \sum_{\nu\geq 1} L_\nu (1-\epsilon^{\nu})\Delta_\infty + a_L \lambda_1 \epsilon Q + O(\epsilon^2),\end{aligned}$$

because $L_1 = L'(0) = a_L \lambda_1$.

Now, we can combine these expansions with (5B.2) to estimate P_{i+1} given that $P_i = (1-\epsilon)\Delta_\infty + \epsilon Q$. Working through the details shows that P_{i+1} is given by

$$\big((1-O(\epsilon))\, F_0 + c_F a_L \lambda_1 \rho'(1)\epsilon Q\big) \otimes \big((1-O(\epsilon))\,\Delta_\infty + \lambda_1 \Delta_0 + \lambda_2 \rho'(1)\epsilon Q\big) + O(\epsilon^2),$$

which can be simplified to

$$(1-O(\epsilon))\,\Delta_\infty + (1-O(\epsilon))\,\lambda_1 F_0 + (1-O(\epsilon))\,\lambda_2\rho'(1)\epsilon Q \otimes F_0 + c_F a_L \lambda_1^2 \rho'(1)\epsilon Q. \quad \text{(5B.3)}$$

Using this, we see that the $f(0) = 0$ condition for the GEC is very similar to the $F_0 = \Delta_\infty$ condition for general channels. In this case, only the last term of (5B.3) matters and an approximate stability condition is given by $c_F a_L \lambda_1^2 \rho'(1) < 1$. If $F_0 \neq \Delta_\infty$, then we must have $\lambda_1 = 0$ so that the second term of (5B.3) vanishes. When $\lambda_1 = 0$, only the third term of (5B.3) remains and the stability condition is given by

$$\lambda_2 \rho'(1) \int_{-\infty}^{\infty} F_0(x) e^{-x/2} dx < 1.$$

We note that the integral in this equation follows from the stability condition derived for memoryless channels in [13], and the symmetry of $F_0(x)$ implied by the random coset assumption.

Bibliography

[1] D. Arnold and H. Loeliger. On the information rate of binary-input channels with memory. In *Proc. IEEE Int. Conf. Commun.*, pages 2692–2695, Helsinki, Finland, June 2001.

[2] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Trans. Inform. Theory*, 20(2):284–287, March 1974.

[3] S. Chung, G. D. Forney, Jr., T. J. Richardson, and R. L. Urbanke. On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit. *IEEE Commun. Letters*, 5(2):58–60, Feb. 2001.

[4] C. Douillard, M. Jézéquel, C. Berrou, A. Picart, P. Didier, and A. Glavieux. Iterative correction of intersymbol interference: Turbo equalization. *Eur. Trans. Telecom.*, 6(5):507–511, Sept. – Oct. 1995.

[5] R. G. Gallager. *Low-Density Parity-Check Codes*. The M.I.T. Press, Cambridge, MA, USA, 1963.

[6] J. Hou, P. H. Siegel, and L. B. Milstein. The performance analysis of low density parity-check codes on Rayleigh fading channels. In *Proc. 38th Annual Allerton Conf. on Commun., Control, and Comp.*, volume 1, pages 266–275, Monticello, IL, USA, Oct. 2000.

[7] J. Hou, P. H. Siegel, L. B. Milstein, and H. D. Pfister. Multilevel coding with low-density parity-check component codes. In *Proc. IEEE Global Telecom. Conf.*, pages 1016–1020, San Antonio, Texas, USA, Nov. 2001.

[8] A. Kavčić, X. Ma, and M. Mitzenmacher. Binary intersymbol interference channels: Gallager codes, density evolution and code performance bounds. submitted to *IEEE Trans. Inform. Theory*, 2001.

[9] B. M. Kurkoski, P. H. Siegel, and J. K. Wolf. Joint message-passing decoding of LDPC codes and partial-response channels. *IEEE Trans. Inform. Theory*, 48(6):1410–1422, June 2002.

[10] M. G. Luby, M. Mitzenmacher, and M. A. Shokrollahi. Analysis of random processes via and-or tree evaluation. In *SODA: ACM-SIAM Symposium on Discrete Algorithms*, pages 364–373, Jan. 1998.

[11] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman, and V. Stemann. Practical loss-resilient codes. In *Proc. of the 29th Annual ACM Symp. on Theory of Comp.*, pages 150–159, 1997.

[12] H. D. Pfister, J. B. Soriaga, and P. H. Siegel. On the achievable information rates of finite state ISI channels. In *Proc. IEEE Global Telecom. Conf.*, pages 2992–2996, San Antonio, Texas, USA, Nov. 2001.

[13] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke. Design of capacity-approaching irregular low-density parity-check codes. *IEEE Trans. Inform. Theory*, 27(1):619–637, Feb. 2001.

[14] V. Sharma and S. K. Singh. Entropy and channel capacity in the regenerative setup with applications to Markov channels. In *Proc. IEEE Int. Symp. Information Theory*, page 283, Washington, DC, USA, June 2001.

[15] M. A. Shokrollahi. New sequences of linear time erasure codes approaching the channel capacity. In *Applicable Algebra in Eng., Commun. Comp.*, pages 65–76, 1999.

[16] M. A. Shokrollahi. Capacity-achieving sequences. In *Codes, Systems, and Graphical Models*, volume 123 of *the IMA Vol. in Math. and its Appl.*, pages 153–166. Springer, 2001.

[17] S. ten Brink. Rate one-half code for approaching the Shannon limit by 0.1 dB. *Electronic Letters*, 36(15):1293–1294, July 2000.

[18] S. ten Brink. Convergence behavior of iteratively decoded parallel concatenated codes. *IEEE Trans. Commun.*, 49(10):1727–1737, Oct. 2001.

[19] S. ten Brink. Exploiting the chain rule of mutual information for the design of iterative decoding schemes. In *Proc. 39th Annual Allerton Conf. on Commun., Control, and Comp.*, Oct. 2001.

[20] R. Urbanke. Iterative coding systems. http://www.calit2.net/events/2001/courses/ics.pdf, Aug. 2001.

[21] N. Varnica and A. Kavčić. Optimized LDPC codes for partial response channels. In *Proc. IEEE Int. Symp. Information Theory*, page 197, Lausanne, Switzerland, June 2002. IEEE.