UNIVERSITY OF CALIFORNIA

Los Angeles

Coding Theoretic Techniques
for Analysis and Mitigation of the Effects of Noise on Algorithms

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Electrical Engineering

by

Shahroze Humayun Kabir

2017

ABSTRACT OF THE THESIS

Coding Theoretic Techniques
for Analysis and Mitigation of the Effects of Noise on Algorithms

by

Shahroze Humayun Kabir

Master of Science in Electrical Engineering
University of California, Los Angeles, 2017
Professor Lara Dolecek, Chair

Coding theory provides techniques which can ensure the error free transmission and storage of data. This data is often used as input to various algorithms that run on hardware. Any information about the algorithm can be useful in helping augment coding theory techniques to protect the data. This thesis studies the performance of two algorithms under noise and how coding theory techniques might be used to mitigate the effects of noise.

The first part of the thesis discusses the effects of noise on the decoding hardware. In particular, the effect of noise due to radiation on low density parity code (LDPC) decoders is studied. The arrival and duration of errors induced by radiation events is also modeled. We accomplish this by proposing a multi-state radiation channel. This model accounts for the duration and dependence of the noise due to radiation. This model also subsumes some previously studied cases and allows for a more refined analysis. We introduce a corresponding LDPC combined Gallager B/E decoder and perform a density evolution analysis to characterize the idealized decoder performance. We also present results in the finite length case.

The second part of the thesis discusses the effects of noisy feature data on the performance of the linear regression algorithm. Machine learning requires a large amount of data to train the learning algorithms. This data must be protected from noise when it is stored or transmitted. Until now, most techniques protect the data agnostic of the application for which the data is to be use. We study the effects of Gaussian noise in the feature data on the output of linear regression. We present coding theoretic techniques to reduce the effects of Gaussian noise on the output of the regression algorithm. We use the expected square loss to measure the effects of noise on the output of regression using repetition coding. We present a technique to optimally allocate units of redundancy to different features to minimize the expected loss given the regression coefficients. We also use submodular optimization to jointly optimize the regression parameters and redundancy allocation at the training stage of regression algorithm. We demonstrate the advantage of our technique in optimizing the redundancy allocation for protecting features.

To my parents.

Table of Contents

List of Figures

CHAPTER 1

Introduction

This thesis primarily analyzes the effects of noise on two data processing algorithms, namely the decoding algorithm for binary Low Density Parity Check (LDPC) codes and the linear regression algorithm. For the LDPC decoder, we look at the effects of noise due to radiation on the performance of the decoding algorithm. For linear regression, we study the effect of Gaussian noise on the data for which the algorithm must make a prediction. We introduce a metric called the Expected Noise Loss to measure the effect of noisy data on the regression algorithm. We present an efficient method to calculate this loss. We further present techniques to create the optimal repetition code given a redundancy budget to minimize this metric. We also introduce a technique to jointly optimize the regression coefficients and the redundancy allocation to each feature in the dataset while training the model. In this chapter we discuss prior works that analyze the effects noise on these algorithms and outlines the contributions in this thesis.

1.1 Effects of radiation induced noise on decoding algorithms

Flash memory devices are increasingly being used in deep-space missions as on-board data storage in spacecraft. On-board storage is necessary to store results, observations, and other data. One example is the Mars Science Laboratory (MSL, also known as the Curiosity Rover) which has an 8 Gb flash chip for its mast camera[1]. The harsh environment these missions take place in involves high levels of radiation which can cause decoding circuitry failures for the device error-correction module. For this reason, the decoder must be robust to radiation-induced errors.

Previous works have modeled the errors caused by exposure to radiation in different ways. Exposure of data storing cells in flash memory to radiation has been modeled as another source of noise in the memory. The authors in [3, 4, 5] discuss codes for this model. The authors in [6] and [7] also consider general asymmetric noise in flash memories.

The operation of the decoder is critical to the performance of the device. As such, decoder failures due to radiation are more dangerous than individual cell errors. Early studies on noisy decoding were made by John von Neumann; he proposed the Triple Modular Redundancy (TMR) scheme. More recently, noisy LDPC decoding has been studied in several papers such as [8, 9, 10, 11, 12, 13] and others. The authors in [14, 15, 16, 17] study noisy decoders with respect to data dependency and energy optimization. A study of noisy decoding for spatially coupled LDPC codes has been performed in [18].

However, previous works do not study noise in the context of radiation errors. The dominant assumption in these related works is that each component is independently affected by noise; the output of each component is passed through an independent, memory-less channel. Usually this channel is assumed to be a binary symmetric channel (BSC). In the context of LDPC decoders, the outputs of the variable and check nodes are passed through corresponding BSCs before being sent to the next node.

This popular approach for dealing with hardware noise does not yield a suitable model for radiation errors. There are two reasons for this. Firstly, errors due to radiation in different components may not be independent. This can be illustrated using Figure 1.1. The heavy-ion track radiation is wide enough to affect several components of the memory device. Thus the errors in these components will not be independent. The authors in [19] confirm that radiation events can affect multiple components simultaneously. Secondly, radiation events can have effects that may be transient such as those caused by single event upsets (SEU). They may also be long lasting such as those caused by total ionizing dosage (TID)

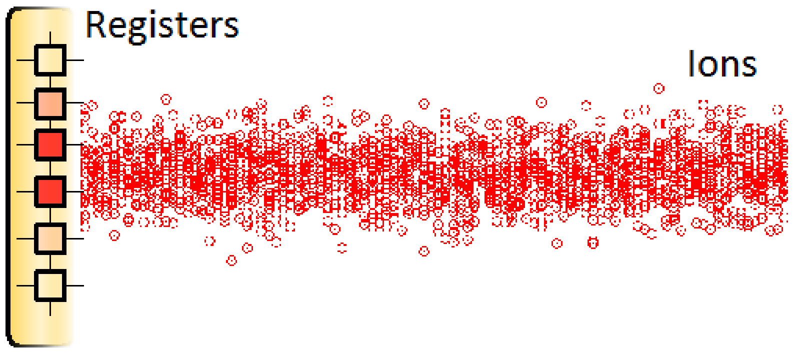

Figure 1.1: Heavy ion track affecting multiple registers.

[19, 20, 21]. As such, there needs to be a notion of duration to study the effects of radiation on decoding circuitry.

1.2 Effects of noise on learning algorithms

The recent surge in the popularity of machine learning is a testament to the power and flexibility of the techniques the field offers. A particularly exciting application of machine learning relates to low-power devices, energy-efficient sensors, wireless networks, etc. For example, such devices are involved in localization and object tracking, security and intrusion detection, media access control, and fault detection.

Low-power devices collect data and transmit it to a central processor (sometimes called the fusion center) where the machine learning models are trained and implemented. Due to the low-power, cost efficient nature of the devices, the data collection process along with the transmission link tend to be noisy. As a result, the algorithm performed at the central processor operates on noisy data. The purpose of this paper is to detail how to tackle this noise.

The traditional approach to dealing with noise involves applying error-correcting codes. In our scenario, however, we face several concerns:

- Low-power and low-cost devices may not admit a powerful (energy-consuming) error-correction architecture,
- The measurements available at the sensor may already be noisy (e.g., due to measurement noise, limited resolution sampling, etc.) and thus the noiseless data is not available for encoding, and
- Classically, the goal is to protect all of the data equally well, but a machine learning model may be influenced by some parts of the data far more than others, so that traditional coding methods are not efficient.

We briefly discuss each of these issues and the role they play in the solution we propose in this paper. We begin with the last concern. The idea here is that error-correction typically seeks to protect all of the data. The application of this data is not considered in the error protection strategy; in fact, the error-correction strategy typically lies in an entirely different abstraction layer. Such an approach is sufficient for general-purpose devices, but in our case, it is not efficient.

Consider, for example, a model with tens of thousands of binary features. A small number of these features may have a very large impact on the model output; such features require significant protection from even a small amount of noise. On the other hand, the vast majority of features may only have a very small impact on the final output, so that these features are resilient to noise. Clearly, a different coding strategy should be implemented for the two sets of features.

Our goal in this work is to introduce a coding strategy that takes the machine learning algorithm into account. We assume that the fusion center has a high-quality model that has access to noiseless training data; however, this model will operate on noisy test data transmitted by the remote low-power measurement devices[1]. The noise in the test data is

[1]More precisely, we consider two cases. In the first case, the high-quality model is fixed and we can only

assumed to be Gaussian in nature as the noise introduced at the output of these low power sensors is often modeled as Gaussian. We seek to develop schemes that tailor the error-protection strategy used by these remote sensors in order to minimize the noise impact on the fusion center's model output.

What type of error-correcting codes suit our needs? Here we must recall the first and second issues described above. Firstly, low-cost devices with energy constraints imply that we must use simple, easily-decoded classes of codes. Secondly, the fact that the noiseless data is not available suggests the use of replication coding. With replication (repetition), the processor can simply ask for more samples as an encoding technique, never relying on the (unavailable) noiseless data. We will attempt to apply these principles to designing codes for protecting linear regression test data.

Related work

There is an abundance of papers studying robust machine learning. For example, the authors in [27] study the experimental performance of several algorithms with artificially noisy datasets. Machine learning algorithms also have to deal with either missing features or features that have been corrupted, the effects of which were studied in [28]. In [29], a game theoretic approach to avoid an over reliance on particular features was discussed. However these papers do not consider coding strategies for improving the performance of the algorithms. In contrast, we use coding theory strategies to protect the data in our application.

Research has also been done based on measuring the "correctness" of the output of the algorithm when the data is known to be noisy. The authors in [30] define the notion of "same decision probability" in order to measure the probability of making the same decision for Bayesian networks when additional information is known. In [31] the authors study

change the error protection for the noisy features. Analyzing this helps us to build up to the second, more novel scenario. In the second case, the model is jointly trained from the noiseless training data and the test data noise characteristics in order to minimize the expected loss when using noisy data. Here too, we further reduce the impact of noise by optimizing the error protection on the test features.

classification techniques when the labels are known to be noisy. Our own work measure the importance of features in terms of how much redundancy needs to be allocated to them.

On the other hand, there is a rich body of work regarding protection of data from different types of noise using coding techniques. This involves coding for data storage mediums such as disk drives[32], flash memory devices[33], non volatile memories[34] and solid state drives[35]. There are also papers proposing coding to improve the robustness of wireless sensor networks [36], [37]. However, as mentioned the techniques in these papers do not take advantage of the extra application-specific information that may be known.

There has been a rising interest regarding developing application-specific coding techniques to solve particular problems. For example, in the problem posed by the authors in [38] for a distributed learning system, some nodes communicate to the central fusion center slower than others. The authors in [38] provide a method called Gradient Coding which involves transmitting a linear combination of data from each node to counteract the effects of some nodes lagging behind others. The authors in [39] study codes for distributed coding. They introduce techniques that are able to recover lost data when a storage node goes down in a distributed storage system. Our own work is in the same vein as these papers as we attempt to address the problem of noisy data for linear regression with coding theory strategies.

This paper builds on our previous work presented in [24], [25] and [26]. In [24], we used an approximation of the mean absolute error (MAE) as the metric to optimize via repetition coding. In [25] and [26], we presented an efficient way of computing the approximate mean absolute loss for linear classifiers and present submodular techniques for optimizing redundancy allocation. In contrast, in this paper, we presents techniques to optimize the actual mean squared loss for regression. We also present a technique to optimize the allocation of redundancy during training, a novel and previously not considered idea.

1.3 Outline of Contributions

A brief outline containing the contributions of this thesis is presented below. The first part of the work deals with analyzing the effects of radiation induced noise on the Gallager B/E decoder. The second part is concerned with analyzing the effects of noisy feature data on the performance of linear regression models. We conclude in chapter 5.

Chapter 2 deal with the effects of radiation on LDPC decoding. We introduce and describe the model for radiation events that takes account of both dependence and duration. We also show how this model applies to the LDPC decoder. We then present the theoretical analysis of the noisy LDPC decoder under our noise framework. This is done using density evolution. We present experimental results to support the analysis obtained using the density evolution analysis.

Chpater 3 deals with coding for linear regression. We present how we model the effects of Gaussian noise on the feature data for linear regression. We demonstrate the use of replication coding. Next, we present our metric to measure the performance of the algorithm and how to optimize a redundancy allocation given regression coefficients using convex and submodular optimization techniques. Building on this, we present how to jointly optimize the linear regression model (that is, the regression coefficients) and redundancy allocation during the training phase using submodular optimization techniques. Finally we present experimental results to show the effectiveness of our techniques in allocating the redundancy budget to different features.

CHAPTER 2

Radiation induced noise on LDPC decoders

In this chapter we analyze the effects of noise due to radiation events on the LDPC decoding algorithm. We develop a probabilistic model for noise which accounts for the duration and distribution of noise events on the decoder. We then analyze the performance of a Gallager B/E decoder(which is Gallager B which can also handle erasures), where the decoder messages are corrupted with this noise model.

2.1 Noise model

In this section we describe the how the noise due to radiation is modeled. Our new model is shown in Figure 2.1. The variable nodes receive the initial estimates from the memory cells. The nodes are connected to the radiation channel. For this analysis, radiation is applied to both the variable nodes and the check nodes.

The radiation channel itself is a two state channel to account for different nodes being affected by radiation events. In the figure, the nodes colored red have been affected by a radiation induced event while the nodes colored black are unaffected. The nodes that operate normally see state A of the channel, which is noiseless. When a node is affected by radiation, the channel state transitions from state A to state B. For a variable node in state B, the channel is a two input three output channel, where the probability of bit erasure is σ_1 and the probability of bit error is σ_2. For the channel, the input symbols are $+1$ (to represent binary 0) and -1 (to represent binary 1), while the output has the additional 0 symbol, to

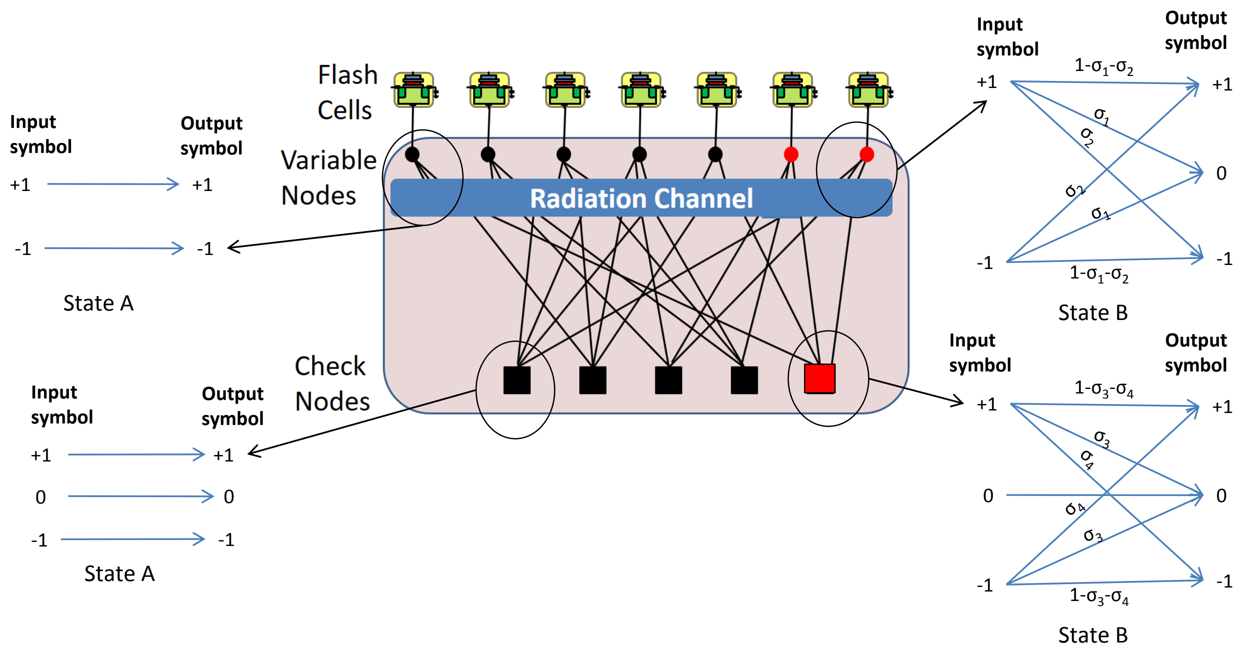

Figure 2.1: Model of the noisy decoder.

represent an erasure. Similarly for a check node in state B, the channel is a three input three output channel, where the probability of bit erasure is σ_3 and the probability of bit error is σ_4. Note that the check node can take an erasure as an input. The erasure probability models the effect of a detectably incorrect voltage level due to radiation, while the error probability models the effect of radiation on a register. The parameters σ_1, σ_2, σ_3 and σ_4 are set according to the strength of the radiation event. Note that multiple nodes can be affected by radiation at the same time in this model, as shown in Figure 2.1.

2.1.1 Poisson process as an event model

Each radiation event affects some number of decoder components. For this analysis, we fix this number as a fraction of the total number of components. In the case of LDPC decoding, if the code length is n, each radiation event affects αn variable nodes and βn check nodes. We select α, β such that αn and βn are non-negative integers. For convenience, we let $1/\alpha, 1/\beta$ be integers as well. Note that α and β are related by the formula $\beta = \frac{d_v}{d_c}\alpha$ in the case of a regular code with variable and check node degrees given by d_v and d_c, respectively, and $\beta = (1-r)\alpha$ for an irregular code with design rate r.

To model how different nodes transition from the noiseless state to the noisy state and back, we propose a queue-based model for the arrival and departure of radiation events. For our analysis we fix the number of components one radiation event affects. For the LDPC decoder, a radiation event can affect a fraction α of the total number of variable nodes in the decoder. The arrival and duration of the events follow the M/M/∞ queue. This models the arrival of radiation events using a Poisson process of rate λ, while the duration is modeled using a process of rate μ. For simplicity, we do not consider the case where a node is affected by multiple events. For the M/M/∞ queue, the probability of j events (for $j \geq 0$) at time ℓ (measured in terms of decoding iterations) is given by [22]:

$$p_j(\ell) = \exp\left(-\frac{\lambda}{\mu}(1-e^{-\mu\ell})\right) \frac{\left(\frac{\lambda}{\mu}(1-e^{-\mu\ell})\right)^j}{j!}, \tag{2.1}$$

There can be at most $1/\alpha$ events active in the decoder, after which the nodes are saturated by radiation. Therefore, the probability is actually given by the following equation:

$$PR_j(\ell) = \begin{cases} p_j(\ell) & 0 \leq j < \ 1/\alpha, \\ \sum_{i=1/\alpha}^{\infty} p_i(\ell) & j = 1/\alpha \end{cases} \tag{2.2}$$

2.1.2 Gallager B/E Decoder

Now we describe the decoder that we will analyze. Each check node generates a message $m_{c \to v}$ sent to each neighboring variable node. Consider a check node c_s with degree d_c. If we generically label the incoming messages from neighboring variable nodes as $m_1, m_2, \ldots, m_{d_c}$, the outgoing message (prior to passing through the radiation channel) to the neighboring variable node v_r is given by the equation:

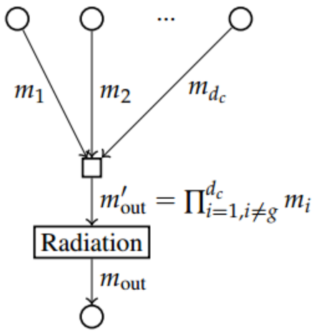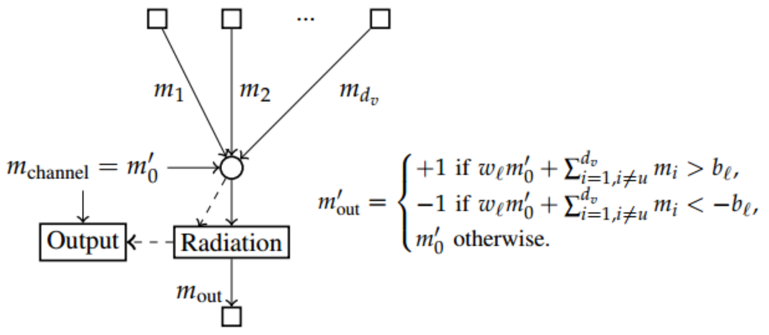
m_1
m_2
m_{d_c}
$m'_{\text{out}} = \prod_{i=1,i\neq g}^{d_c} m_i$
Radiation
m_{out}
m_1
m_2
m_{d_v}
$m_{\text{channel}} = m'_0$
Output
Radiation
m_{out}
$m'_{\text{out}} = \begin{cases} +1 \text{ if } w_\ell m'_0 + \sum_{i=1,i\neq u}^{d_v} m_i > b_\ell, \\ -1 \text{ if } w_\ell m'_0 + \sum_{i=1,i\neq u}^{d_v} m_i < -b_\ell, \\ m'_0 \text{ otherwise.} \end{cases}$

Figure 2.2: Check node message to the g-th neighboring variable node (left) and variable node message to the u-th check node (right).

$$m'_{c_s \to v_r} = \prod_{j=1,j\neq r}^{d_c} m_j. \tag{2.3}$$

Now we define the variable node operation. For a variable node v_t of degree d_v, with incoming messages from neighboring check nodes generically labeled $m_1, m_2, \ldots, m_{d_v}$ and a message from the channel m'_0, the message $m'_{v_t \to c_u}$ is generated and sent to the check node c_u. Recall that ℓ is the iteration index. This message is generated using the following equation:

$$m'_{v_t \to c_u} = \begin{cases} +1 \text{ if } (w_\ell m'_0 + \sum_{j=1,j\neq u}^{d_v} m_j) > b_\ell, \\ -1 \text{ if } (w_\ell m'_0 + \sum_{j=1,j\neq u}^{d_v} m_j) < -b_\ell, \\ m'_0 \text{ otherwise.} \end{cases}$$

Here w_l is the weight assigned to the channel message m'_0 and b_l is the voting threshold. Both w_l and b_l are greater than or equal to 0. The message $m'_{v_t \to c_u}$ is passed through the radiation channel and then sent to the check node. The final estimate involves a similar computation by considering messages from all incoming check nodes. Figure 2.2 shows an illustration of a particular instance of the check and variable message calculation. In the figure, the variable node(right) is generating a message for its u-th check node neighbor, while the check node(left) is generating a message for its g-th variable node neighbor. This model can now subsume previous models by changing relevant parameters. For example the model in [10] can be implemented by setting $\sigma_1 = 0$, $w_l = 0$, $\alpha = 1$, $\lambda \to \infty$, $\mu = 0$ and $\sigma_2 = p_{MAJ}$.

2.2 Density evolution and analysis

In this section we perform density evolution analysis. This analysis helps in determining theoretical limits of our LDPC decoder. One important observation is that, due to the radiation in the final read operations of the decoder, the residual bit error rate (RBER) will be non-zero. Furthermore, unlike the conventional model, the performance analysis may not necessarily be an upper bound for the finite length case. This is because a practical decoder may stop decoding after successive error-free iterations. The arguments and requirements for density evolution are as described in [11]. Symmetry allows us to analyze the case of all +1 symbol codeword. The density evolution analysis involves tracking probabilities of variable to check and check to variable messages at each iteration.

Claim 1. *Consider a check (variable) node of degree d_c (d_v). Let $\gamma_{j,k}$ ($\delta_{j,k}$) represent the probability that k of the $d_c - 1$ variable nodes (k of the $d_v - 1$ check nodes) incident to the check (variable) node are affected by radiation given j radiation events according to our model. Then,*

$$\gamma_{k,j} = \binom{d_c - 1}{k} (j\alpha)^k (1 - j\alpha)^{d_c - 1 - k},$$

and,

$$\delta_{k.j} = \binom{d_v - 1}{k} (j\beta)^k (1 - j\beta)^{d_v - 1 - k}.$$

Proof. First, we have that $j\alpha n$ total variable nodes are affected by radiation, so that the probability of selecting a variable node affected by radiation is $j\alpha$. The remainder of the claim is a consequence of averaging over the entire ensemble and taking the limit of the block length n to infinity. The proof for $\delta_{k,j}$ follows the same idea. $\square$

Consider a code of length n with the fraction of edges connected to variable nodes of degree a ($2 \leq a \leq v_{max}$) denoted by τ_a^v and the fraction of edges connected to check nodes of degree b ($2 \leq b \leq c_{max}$) denoted τ_b^c. We do not allow for variable or check nodes of degree 1 or less. In the case of regular codes, we have that $\tau_{d_v}^v = \tau_{d_c}^c = 1$ for some d_v and d_c.

We denote by $\rho_+^{(\ell)}, \rho_0^{(\ell)}$, and $\rho_-^{(\ell)}$ the probability that a variable to check message $m_{v\to c}$ at the ℓth iteration is equal to $+1, 0$, and -1, respectively. These probabilities are influenced by the radiation channel and its inputs; the probabilities of message values leaving variable nodes but prior to the application of the radiation channel are given by $\rho_+^{\prime(\ell)}$ and $\rho_-^{\prime(\ell)}$ for $+1$ and -1, respectively. The probabilities for check to variable node messages $m_{c\to v}$ are given by $q_+^{(\ell)}, q_0^{(\ell)}$, and $q_-^{(\ell)}$ for $+1, 0$, and -1, respectively. These probabilities are also influenced by the radiation channel and inputs; the hypothetical probabilities of hypothetical message values leaving check nodes but before radiation are given by $q_+^{\prime(\ell)}, q_0^{\prime(\ell)}, q_-^{\prime(\ell)}$ for $+1, 0, -1$, respectively. Let us first express the $\rho^{(\ell)}$ and $q^{(\ell)}$ probabilities in terms of the $\rho^{\prime(\ell)}$ and the $q^{\prime(\ell)}$. We have the following fact.

Fact 1. *For variable-to-check messages affected by radiation (that is, those messages passed through a radiation channel in state B as depicted in Figure 2.1.),*

$$\begin{aligned}
\rho_+^{(\ell)} &= (1-\sigma_1-\sigma_2)\rho_+^{\prime(\ell)} + \sigma_2\rho_-^{\prime(\ell)}, \\
\rho_0^{(\ell)} &= \sigma_1(\rho_+^{\prime(\ell)} + \rho_-^{\prime(\ell)}) = \sigma_1, \\
\rho_-^{(\ell)} &= \sigma_2\rho_+^{\prime(\ell)} + (1-\sigma_1-\sigma_2)\rho_-^{\prime(\ell)}.
\end{aligned}$$

For check-to-variable messages affected by radiation,

$$\begin{aligned} q_+^{(\ell)} &= (1-\sigma_3-\sigma_4)q_+'^{(\ell)} + \sigma_4 q_-'^{(\ell)}, \\ q_0^{(\ell)} &= \sigma_3(q_+'^{(\ell)} + q_-'^{(\ell)}) + q_0'^{(\ell)} = \sigma_3 + (1-\sigma_3)q_0'^{(\ell)}, \\ q_-^{(\ell)} &= \sigma_4 q_+'^{(\ell)} + (1-\sigma_3-\sigma_4)q_-'^{(\ell)}. \end{aligned}$$

For messages not affected by radiation, $\rho_+^{(\ell)} = \rho_+'^{(\ell)}, \rho_0^{(\ell)} = 0$, $\rho_-^{(\ell)} = \rho_-'^{(\ell)}$, $q_+^{(\ell)} = q_+'^{(\ell)}, q_0^{(\ell)} = q_0'^{(\ell)}$, *and* $q_-^{(\ell)} = q_-'^{(\ell)}$.

Proof. Follows directly from the radiation channels shown in Figure 2.1. □

We use the following well-known facts:

Fact 2. *For* m *a non-negative integer,*

$$\sum_{i\geq 1 \ odd}^{m} \binom{m}{i} x^i z^{m-i} = \frac{1}{2}\left[(x+z)^m - (-x+z)^m\right],$$

and

$$\sum_{i\geq 0 \ even}^{m} \binom{m}{i} x^i z^{m-i} = \frac{1}{2}\left[(x+z)^m + (-x+z)^m\right].$$

To keep our expressions clean, we define

$$O_m(x,z) := \frac{1}{2}\left[(x+z)^m - (-x+z)^m\right] \text{ and } E_m(x,z) := \frac{1}{2}\left[(x+z)^m + (-x+z)^m\right].$$

Our first result is a recursive expression for $q_+^{(\ell)}, q_0^{(\ell)}$, and $q_-^{(\ell)}$.

Lemma 1. *For* $\ell \geq 1$, *the check to variable node probabilities for a check node of degree* d_c

satisfy

$$q_+^{\prime(\ell),j,d_c} = \sum_{k=0}^{d_c-1} \gamma_{k,j}(E_k(\rho_-^{(\ell)}, \rho_+^{(\ell)}) E_{d_c-k-1}(\rho_-^{\prime(\ell)}, \rho_+^{\prime(\ell)}) + O_k(\rho_-^{(\ell)}, \rho_+^{(\ell)}) O_{d_c-k-1}(\rho_-^{\prime(\ell)}, \rho_+^{\prime(\ell)})),$$

$$q_-^{\prime(\ell),j,d_c} = \sum_{k=0}^{d_c-1} \gamma_{k,j}(E_k(\rho_-^{(\ell)}, \rho_+^{(\ell)}) O_{d_c-k-1}(\rho_-^{\prime(\ell)}, \rho_+^{\prime(\ell)}) + O_k(\rho_-^{(\ell)}, \rho_+^{(\ell)}) E_{d_c-k-1}(\rho_-^{\prime(\ell)}, \rho_+^{\prime(\ell)})), \text{ and}$$

$$q_0^{\prime(\ell),j,d_c} = \sum_{k=0}^{d_c-1} \gamma_{k,j}(1-(1-\sigma_1)^k).$$

Let us define

$$S_j(x) = (1 - j\alpha x)^{d_c-1}.$$

Then, the above quantities can be simplified to

$$q_+^{\prime(\ell),j,d_c} = \frac{1}{2} S_j(\sigma_1) + \frac{1}{2} S_j(\sigma_1 + 2\sigma_2)(\rho_+^{\prime(\ell)} - \rho_-^{\prime(\ell)})^{d_c-1}, \tag{2.4}$$

$$q_-^{\prime(\ell),j,d_c} = \frac{1}{2} S_j(\sigma_1) - \frac{1}{2} S_j(\sigma_1 + 2\sigma_2)(\rho_+^{\prime(\ell)} - \rho_-^{\prime(\ell)})^{d_c-1}, \tag{2.5}$$

and

$$q_0^{\prime(\ell),j,d_c} = 1 - S_j(\sigma_1).$$

Proof. The first factor inside the summation is the probability of k radiation-affected nodes given j radiation events, $\gamma_{k,j}$. Now we have k nodes affected by radiation and $d_c - 1 - k$ nodes not affected. For the outgoing message from the check node to be positive, we require either an even number of negative messages from both affected and unaffected variable nodes, contributing the probability term $E_k(\rho_-^{(\ell)}, \rho_+^{(\ell)}) E_{d_c-k-1}(\rho_-^{\prime(\ell)}, \rho_+^{\prime(\ell)})$, or an odd number of negative messages from both the affected and unaffected variable nodes, contributing the probability term $O_k(\rho_-^{(\ell)}, \rho_+^{(\ell)}) O_{d_c-k-1}(\rho_-^{\prime(\ell)}, \rho_+^{\prime(\ell)})$.

Note that since the unaffected variable nodes pass their messages through a noiseless channel, as shown in Figure 2.1, we may write $\rho_-'^{(\ell)}$ and $\rho_+'^{(\ell)}$ instead of $\rho_-^{(\ell)}$ and $\rho_+^{(\ell)}$, respectively. To obtain the simplification, we can explicitly write out the two terms for each of the E_k and O_k expressions and perform the multiplication. The cross terms cancel. This leaves us with

$$\begin{aligned} q_+'^{(\ell),j,d_c} &= \sum_{k=0}^{d_c-1} \gamma_{k,j} \left(\frac{1}{2}(\rho_+^{(\ell)} + \rho_-^{(\ell)})^k (\rho_+'^{(\ell)} + \rho_-'^{(\ell)})^{d_c-k-1} \right. \\ &\qquad \left. + \frac{1}{2}(\rho_+^{(\ell)} - \rho_-^{(\ell)})^k (\rho_+'^{(\ell)} - \rho_-'^{(\ell)})^{d_c-k-1} \right). \end{aligned}$$

Next, we observe that $\rho_+'^{(\ell)} + \rho_-'^{(\ell)} = 1$ and $\rho_+^{(\ell)} + \rho_-^{(\ell)} = ((1-\sigma_1-\sigma_2)+\sigma_2)(\rho_+'^{(\ell)} + \rho_-'^{(\ell)}) = 1-\sigma_1$, using Fact 1. We also have that $\rho_+^{(\ell)} - \rho_-^{(\ell)} = ((1-\sigma_1-\sigma_2)-\sigma_2)(\rho_+'^{(\ell)} - \rho_-'^{(\ell)}) = (1-\sigma_1-2\sigma_2(\rho_+'^{(\ell)} - \rho_-'^{(\ell)})$. Using these simplifications, we obtain

$$\begin{aligned} q_+'^{(\ell),j,d_c} &= \sum_{k=0}^{d_c-1} \gamma_{k,j} \left(\frac{1}{2}(1-\sigma_1)^k + \right. \\ &\qquad \left. \frac{1}{2}(1-\sigma_1-2\sigma_2)^k (\rho_+'^{(\ell)} - \rho_-'^{(\ell)})^{d_c-1} \right). \end{aligned} \tag{2.6}$$

Now, we note that

$$\begin{aligned} &\sum_{k=0}^{d_c-1} \gamma_{k,j}(1-x)^k \\ &\quad = \sum_{k=0}^{d_c-1} \binom{d_c-1}{k} (j\alpha)^k (1-j\alpha)^{d_c-1-k} (1-x)^k \\ &\quad = \sum_{k=0}^{d_c-1} \binom{d_c-1}{k} (1-j\alpha)^{d_c-1-k} (j\alpha - j\alpha x)^k \\ &\quad = (1-j\alpha x)^{d_c-1} = S_j(x). \end{aligned}$$

Here, in the third step we used the binomial theorem. Next, using the formula $S_j(x) = \sum_{k=0}^{d_c-1} \gamma_k (1-x)^k$ in (2.6), we get

$$q_+'^{(\ell),j,d_c} = \frac{1}{2} S_j(\sigma_1) + \frac{1}{2} S_j(\sigma_1 + 2\sigma_2)(\rho_+'^{(\ell)} - \rho_-'^{(\ell)})^{d_c-1}.$$

Similarly, for the outgoing message from the check node to be negative, we require either an odd number of incoming negative messages that are radiation affected and an even number of unaffected negative messages, or, an even number of incoming affected negative messages and an odd number of unaffected negative messages. These two possible cases make up the expression for $q_-^{(\ell)}$. Simplifying, we can write

$$q_-^{\prime(\ell),j,d_c} = \sum_{k=0}^{d_c-1} \gamma_{k,j} \left(\frac{1}{2}(\rho_+^{(\ell)} + \rho_-^{(\ell)})^k (\rho_+^{\prime(\ell)} + \rho_-^{\prime(\ell)})^{d_c-k-1} - \frac{1}{2}(\rho_+^{(\ell)} - \rho_-^{(\ell)})^k (\rho_+^{\prime(\ell)} - \rho_-^{\prime(\ell)})^{d_c-k-1} \right),$$

and use the same ideas as the case of $q_+^{\prime(\ell),j,d_c}$ to get our result.

Finally, for an outgoing message to be an erasure, we require at least one of the incoming messages to be an erasure. The probability none of these are erased is given by $(1-\sigma_1^{(\ell)})^k$ (note that the unaffected messages cannot be erased.) This gives the last expression. □

Next, we compute the variable to check probabilities at the $\ell+1$ iteration that are computed using the check to variable probabilities from iteration ℓ. Of course, it is easy to relate these to the radiation-affected probabilities by conditioning over the number of radiation events and applying Fact 1. We write $\bar{\epsilon}$ for $1-\epsilon$.

Lemma 2. *For $\ell = 0$, assuming the all +1's codeword is transmitted (using the $0 \to +1$ and $1 \to -1$ mapping) and ϵ is the channel error probability,*

$$\rho_+^{\prime(0)} = 1 - \epsilon \text{ and } \rho_-^{\prime(0)} = \epsilon.$$

Let

$$Q'(i_1, i_2, i_3) := \left(\sum_{b=2}^{c_{max}} \tau_b^c (q_+^{\prime(\ell),j,b})\right)^{i_1} \left(\sum_{b=2}^{c_{max}} \tau_b^c (q_0^{\prime(\ell),j,b})\right)^{i_2} \left(\sum_{b=2}^{c_{max}} \tau_b^c (q_-^{\prime(\ell),j,b})\right)^{i_3},$$

and

$$Q(i_1, i_2, i_3) :=$$
$$\left(\sum_{b=2}^{c_{max}} \tau_b^c (q_+^{(\ell),j,b})\right)^{i_1} \left(\sum_{b=2}^{c_{max}} \tau_b^c (q_0^{(\ell),j,b})\right)^{i_2} \left(\sum_{b=2}^{c_{max}} \tau_b^c (q_-^{(\ell),j,b})\right)^{i_3}.$$

For $\ell \geq 1$, $\rho_-^{`(\ell+1)} = \sum_{j=1}^{1/\alpha} PR_j(\ell) \rho_-^{`(\ell+1),j}$, *and*

$$\begin{aligned} \rho_-^{`(\ell+1),j} &= \sum_{k=0}^{d_v-1} \delta_{k,j} \times \sum_{a=2}^{v_{max}} \tau_a^v \times \\ &\Bigg(\epsilon\Bigg(\sum_{\substack{i_-,i'_-,i_0,i'_0,i_+,i'_+ \\ -w_\ell+i_++i'_+-i_--i'_- \leq b_\ell}} \binom{k}{i_+, i_0, i_-}\binom{a-k-1}{i'_+, i'_0, i'_-} \times \\ &Q(i_+, i_0, i_-)Q'(i'_+, i'_0, i'_-)\Bigg) + \\ &\bar{\epsilon}\Bigg(\sum_{\substack{i_-,i'_-,i_0,i'_0,i_+,i'_+ \\ w_\ell+i_++i'_+-i_--i'_- < -b_\ell}} \binom{k}{i_+, i_0, i_-}\binom{a-k-1}{i'_+, i'_0, i'_-} \times \\ &Q(i_+, i_0, i_-)Q'(i'_+, i'_0, i'_-)\Bigg)\Bigg). \end{aligned} \tag{2.7}$$

Lastly, $\rho_+^{`(\ell+1)} = 1 - \rho_-^{`(\ell+1)}$, *and* $\rho_+^{`(\ell+1),j} = 1 - \rho_-^{`(\ell+1),j}$.

Note that if the code is regular, with degrees d_v, d_c, the Q' functions simplify to $Q'(i_1, i_2, i_3) := \left(q_+^{\prime(\ell),j,d_c}\right)^{i_1} \left(q_0^{\prime(\ell),j,d_c}\right)^{i_2} \left(q_-^{\prime(\ell),j,d_c}\right)^{i_3}$ (and similarly for Q). Next, we give the proof.

Proof. The result follows from our decoder depicted in Figure 2.2. If we transmit the all $+1$'s codeword, the probability of a particular variable node (prior to radiation application) having initial value 1 is given by $1 - \epsilon$, while the probability of having initial value -1 is given by ϵ.

Next, we apply the rules from the decoder. Let us say we are working with a variable node of degree a, that there are j current radiation events, and that the average number of check nodes affected by the radiation incident to a particular node is k. A value of -1

to be sent from a variable node requires either an initial flip from $+1$ to -1 (probability ϵ) with insufficient check node messages to flip this value back, or no initial flip (probability $1-\epsilon$) with sufficiently many messages to flip it to -1. There are two types of check node messages: those affected by radiation, and those that are not. We write i_+, i_-, i_0 for the messages affected by radiation that are $+1, -1$, and 0 (erasure), respectively. The unaffected messages are written similarly, but with an apostrophe.

In the first case, to avoid flipping to $+1$, we need the condition $w_\ell + i_+ + i'_+ - i_- - i'_- \leq b_\ell$. Recall that our current variable node in the summation has degree a. Our variable node receives $a-1$ messages, of which k are radiation affected and $a-k-1$ are not. We consider all possibilities for the message values. To compute the average probability of an incoming message from a check node with a particular value, we consider the proportion of incoming messages from check nodes of degree τ^c_b and average the q probabilities.

Similarly, if the initial value is 1, (prob. $1-\epsilon$) we need sufficiently many $-1's$ (at least enough to satisfy $w_\ell + i_+ + i'_+ - i_- - i'_- < b_\ell$) to flip it to -1. The probabilities for a $+1$ value at a variable node follow from similar logic. A final average is performed over the variable node degrees, the number of radiation-affected incident nodes k, and the number of radiation events j. □

The above lemma gives the recursive equation used to track variable message probabilities. It is useful to track this quantity as it potentially allows us to find a true threshold, in some cases, for which the decoder probability of error will converge to zero. At the final read operation, each variable node computes the final output by using all the incoming messages. The following quantities track the probabilities for this output:

Definition 1. *Let the auxiliary quantities $\hat{\rho}_-^{\prime(\ell+1)}$ and $\hat{\rho}_-^{\prime(\ell+1)}$ be defined the same as the versions without hats, but using d_v and d_c in place of $d_v - 1$ and $d_c - 1$, respectively.*

Using the above defined quantities, we can now derive the expression for the probability of decoder output error. Note that radiation can effect the output of the decoder at the final read step. We take this into account in our derivation. The equation for the output error is as follows:

$$\rho_E^{(\ell+1)} = \sum_{j \geq 0} PR_j(\ell+1) \left[(1-j\alpha)(\hat{\rho}_-^{\prime(\ell+1)}) + \right.$$
$$\left. (j\alpha)((1-\sigma_1-\sigma_2)\hat{\rho}_-^{\prime(\ell+1)} + \sigma_2(\hat{\rho}_+^{\prime(\ell+1)}) + \sigma_1\epsilon) \right] . \tag{2.8}$$

We show the previous equation in the following way. First, recall that the hat quantities such as $\hat{\rho}_-^{\prime(\ell+1)}$ are analogous to the normal radiation unaffected variable to check node messages probabilities; however, they are computed from all incoming check nodes at each variable node, matching our decoding setup from Figure 2.1. For $\rho_E^{(\ell+1)}$, as usual, we first condition over the total number of radiation events. If there are j such events, the variable node is not affected with probability $1 - j\alpha$, so that the decoder error is simply the probability that a variable node has computed an incorrect message, given by $\hat{\rho}_-^{\prime(\ell+1)}$. If our variable node is affected by radiation, with probability $j\alpha$, there are three possible sources of error. In the first, the variable node has an incorrect message which is not changed by radiation (probability $1-\sigma_1-\sigma_2$). In the second case, the variable node has a correct message flipped (probability σ_2). Lastly, there is an erasure, regardless of the value of the radiation-unaffected message (probability σ_1). In the latter case, we default to using the original message, which has error probability ϵ.

2.2.1 Analysis

We now analyze the results performance of the decoder under various levels of radiation, using the recursive equations derived from the density evolution. We depict the bit error rate for a code with $d_v = 3$ and $d_c = 6$ (design rate 0.5) in Figure 2.3. We show a range of parameters α (the fraction of total nodes affected by one radiation event) and ϵ (the initial channel crossover probability) ranging between 10^{-5} and 10^{-1}. The radiation event process arrival and departure parameters were 0.01 and 0.05, respectively. We also used a radiation

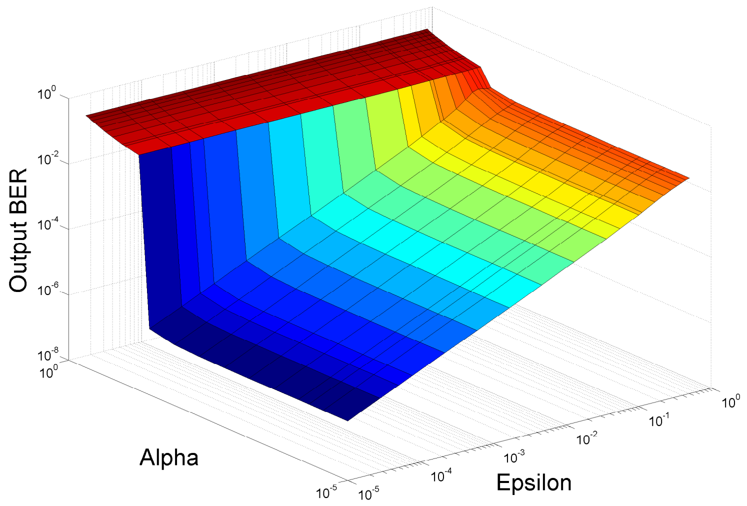

Figure 2.3: Output bit error rate (BER) computed through theoretical evaluation for a (3, 6)-regular code with ϵ and α selected from $[10^{-5}, 10^{-1}]$. We selected $\lambda = 0.01$, $\mu = 0.05$, and $\sigma_1 = \sigma_2 = \sigma_3 = \sigma_4 = 0.25$.

channel with $\sigma_1 = \sigma_2 = = \sigma_3 = \sigma_4 = 0.25$. We used the decoder parameters $b = w = 0$ at all iterations.

The results in Figure 2.3 are fairly representative of the typical output of the theoretical evaluation. First, note that the parameter α (the fraction of nodes affected by radiation in one event) matters more than the channel parameter ϵ. That is, a high α is disastrous for the decoder, while a high ϵ can sometimes be tolerated. Therefore there are two regimes, one where α is the dominant factor controlling the RBER and the second regime where ϵ is dominant.

An interesting observation is that check node noise does not have a significant impact on the overall performance. Changing σ_3 and σ_4 to zero does not significantly change the plot in Figure 2.3. In other words, it is possible to approximate the system by using variable node noise only. We can examine check node noise alone by setting $\sigma_1 = \sigma_2 = 0$ and $\sigma_3 = \sigma_4 = 0.25$. The resulting mesh, shown in Figure 2.4, has similarities in its shape to the VN noise or combined noise mesh, but it has a much lower residual BER. Thus, it is possible to recover from check node errors easier.

Another insight that follows from this analysis is the reason for the non-zero minimum

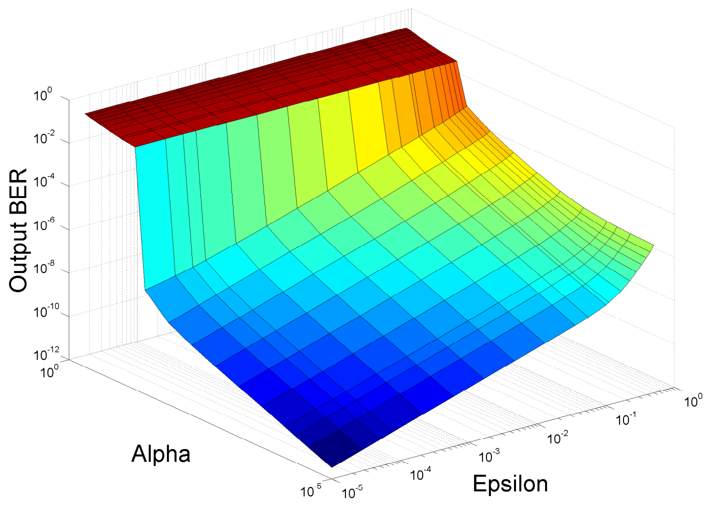

Figure 2.4: Output bit error rate (BER) for check node noise only ($\sigma_1 = \sigma_2 = 0$, $\sigma_3 = \sigma_4 = 0.25$). The other parameters are the same as in Figure 2.3. Note that the output bit error rate reaches as low as 10^{-12}, compared to 10^{-7} in Figure 2.3.

RBER. This can be explained by the fact that a radiation event at the final read operation is possible. In a regime where the decoder performs well, this radiation event at the final read contributes most to the RBER. Even in the case where the decoder performs perfectly, the RBER is non-zero. The minimum RBER can be approximated (using the expectation of the $M/M/\infty$ queue) as follows:

$$\begin{aligned}\rho_E^{(\ell+1)} &= \sum_{j\geq 0} PR_j(\ell+1)(j\alpha)(\sigma_2+\sigma_1\epsilon) \\ &\approx \frac{\lambda}{\mu}(1-e^{-\mu\ell})(\alpha\sigma_2+\alpha\sigma_1\epsilon).\end{aligned} \tag{2.9}$$

Finally we compare our model with the traditional i.i.d. noise model. We plot the decoder error for both models over the same range of values. Naturally, if we simply ignore α (that is, taking $\alpha = 1$) with $\sigma_1 + \sigma_2 = 0.5$, the resulting system will be nearly useless. Consider a less noisy case, where $\sigma_1 = 0$ and $\sigma_2 = 0.01$. In this case, with the same α, λ, μ parameters as chosen above, we have a vastly more precise result as well, shown in Figure 3.4. The colored mesh is the result of our model, while the transparent mesh represents an i.i.d.

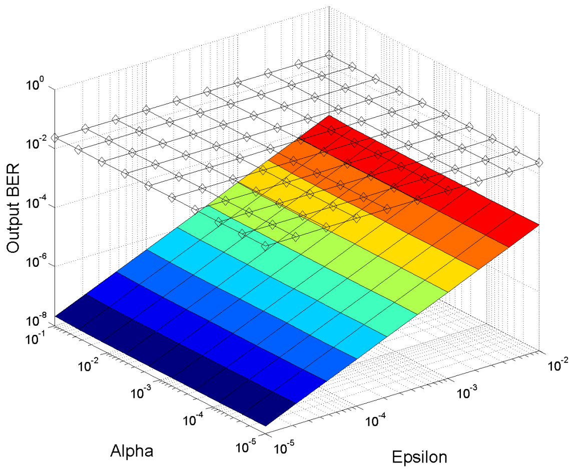

Figure 2.5: Output BER comparison for i.i.d model(transparent mesh) and our model(colored mesh).

noise model. This is an important benefit of our model; the unrefined baseline model leads to results that are too pessimistic, potentially leading to code design that is unnecessarily conservative.

Next, we use our density evolution analysis, to see the effect that the voting threshold b has on the decoder output in terms of the BER. We show this in Figure 2.6. This figure shows the decoder performance for different values of the voting threshold b, for the regular $(4, 8)$ code over a range of values of (which is the number of variable nodes impacted per radiation event). The parameters for this analysis were $\lambda = 0.005$, $\epsilon = 0.006$, $\mu = 0.03$ and $\sigma_1 = \sigma_2 = \sigma_3 = \sigma_4 = 0.25$. We can see that as the value of α increases, the value of b required to obtain the optimal performance in terms of BER increases by 1.

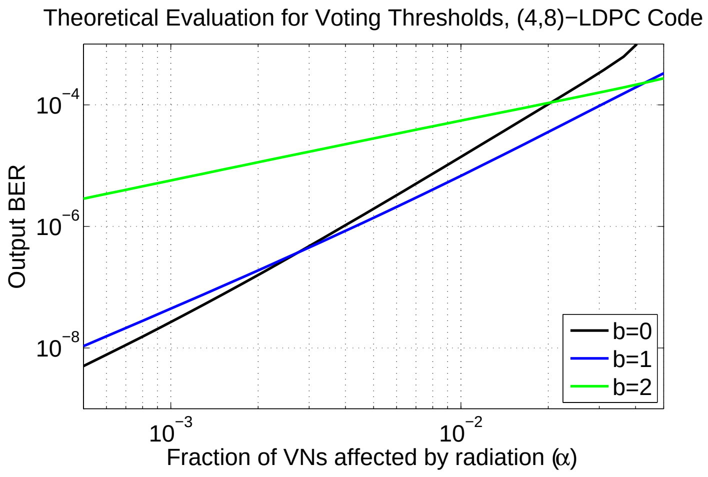

Figure 2.6: Theoretical BER comparison for (4,8) code against α for different values of voting threshold b.

2.3 Experimental Results

In this section we present finite length simulations to support our density evolution analysis. Through finite length simulations, we seek to validate the accuracy and relevance of the theoretical evaluation. In order to simulate code performance, we have fully implemented both the radiation channel model and our Gallager B/E hybrid decoder in MATLAB. It is important to note that our theoretical analysis provides an approximation—not a lower bound—of the residual bit error rate. This approximation is expected because we keep the BSC channel error rates much lower than the decoding threshold of the code/decoder pairs, thus the BER is dominated by the internal noise.

We set $\sigma_1 = \sigma_2 = \sigma_3 = \sigma_4 = 0.25$. Thus any node affected by a radiation event has an equal chance of being in State B or State A, and any variable node in State B has an equal chance of being an erasure or a flipped-bit. We set our decoder to terminate after 30 iterations (if it had not already stopped decoding due to the all 0's syndrome).

We use LDPC codes with length $n = 2048$ bits. We set the initial probability of error in the BSC channel: $\epsilon = 0.004$; on average ≈ 8 variable nodes initially have the incorrect value. The radiation arrival rate λ is the variable in the presented simulation results, varying from

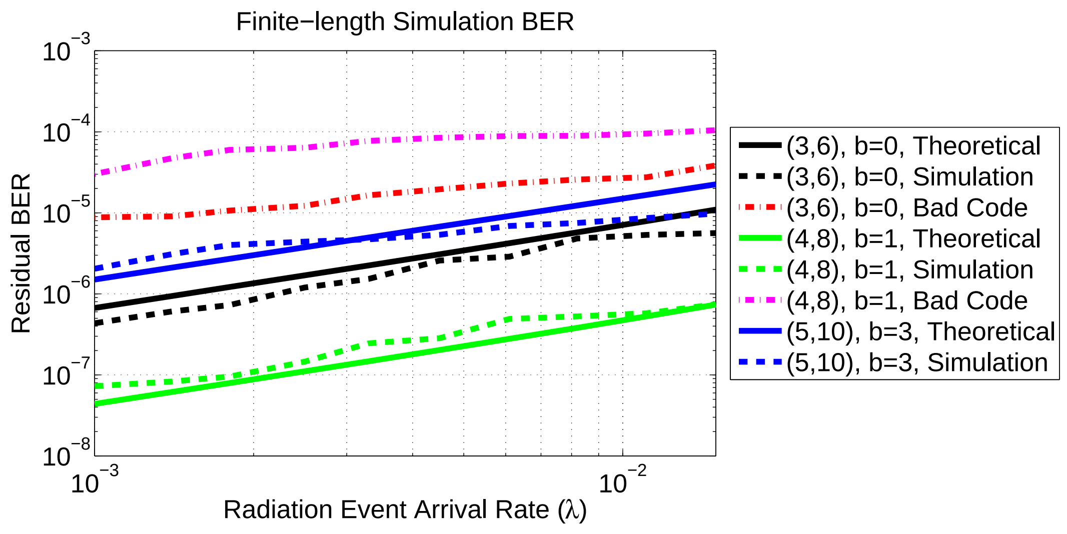

Figure 2.7: Finite-length simulations and their corresponding theoretical evaluations. The noise parameters are as follows: $\epsilon = 0.004, \mu = 0.2, \alpha = 1/2048, \beta = 5/2048$. The dependent variable is the radiation arrival rate, λ, and the y-axis is the residual BER. The solid curves are the BERs from the theoretical evaluations, the dashed lines are the BERs from the finite-length simulation of the good codes with girth 8, and the dotted lines are the BERs from the simulations of the bad codes with girth 4.

0.001 to 0.015. The radiation departure rate is set to $\mu = .2$, meaning it takes an average of 5 iterations for a radiation event to depart. We set $\alpha = 10/2048$ and $\beta = 5/2048$ so that when a radiation even occurs, 10 variable nodes and 5 check nodes are in the radiation state.

In Figure 2.7 we present results over protograph-based regular $(3, 6)$, $(4, 8)$ and $(5, 10)$-LDPC codes that are constructed using the circulant PEG algorithm. For the construction of the $(3, 6)$ code, we start with a 4 variable node, 2 check node protograph. The first variable node is connected to the first check node by two parallel edges and to the second check node by a single edge. All other variable nodes use the same connection configuration with alternation between two check nodes. Next, the protograph is lifted by a factor of 4, using 4x4 circulant permutations to get girth 4. The resulting protograph then is lifted by a factor of 128 using 128x128 circulant permutations. Lastly, the circulant PEG algorithm was used to assign circulant permutations to maximize the girth to 8. The other codes were constructed in a similar fashion and all have rate 1/2 ($n = 2048, k = 1024$). This construction is a known good approach to designing LDPC codes in the noiseless component case [23]. In

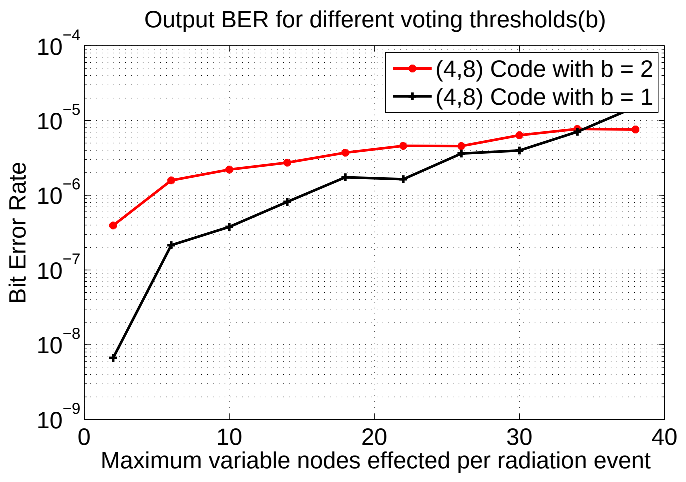

Figure 2.8: Finite lenghth simulation BER comparison for (4,8) code against α for different values of voting threshold b.

addition to the above codes, we also used "bad" codes with girth of 4. As expected, there was a mismatch with the theoretical analysis due to the cycle-free assumption.

Next, we present the experimental performance of the decoder as a function of the number of nodes that are impacted per radiation event. In Figure 2.8, we plot the output BER versus number of nodes impacted per event for a $(4, 8)$-LDPC regular code of length 2048 generated using the method described previously. The remaining parameters were $\lambda = 0.005$, $\epsilon = 0.006$, $\mu = 0.03$ and $\sigma_1 = \sigma_2 = \sigma_3 = \sigma_4 = 0.25$ We can see that it matches the trend shown by the density evolution. Clearly as the nodes impacted per radiation event increases, so does the BER. Another interesting thing note is that as the number of nodes impacted increases, we see that a higher value of the threshold in the decoder provides better performance. This is due to the fact that as more nodes become unreliable due to radiation, a lower threshold means that it is more likely that incorrect messages will cause the variable node to decode incorrectly. This supports the results that were obtained from the density evolution analysis.

CHAPTER 3

Coding for linear regression

In this chapter, we will discuss techniques used to protect test data for linear from Gaussian noise. We also discuss how to jointly optimize the redundancy allocation and regression weights while training the regression model. We begin by establishing notation for linear regression and some preliminaries regarding the noise models used in this paper. We define a feature vector $\mathbf{x} = (x_1, \ldots, x_n)$ where we consider elements $x_j \in \mathbb{F}_2$ (for binary regression) and $x_j \in \mathbb{R}$ (for regression with continuous values). We define the vector of regression coefficients $\mathbf{a} = (a_1, \ldots, a_n)$ where $a_j \in \mathbb{R}$. The target value corresponding to the jth feature vector is given by $y_j \in \mathbb{R}$. We also define the matrix $\mathbf{X}$ of p feature vectors and the corresponding vector $\mathbf{y} \in \mathbb{R}^p$ containing the corresponding p target values. For a given feature vector $\mathbf{x}$, linear regression predicts the value of the target variable y as $\mathbf{a}^T\mathbf{x}$.

The noise model that we consider is shown in Figure 3.1. The noisy version of feature

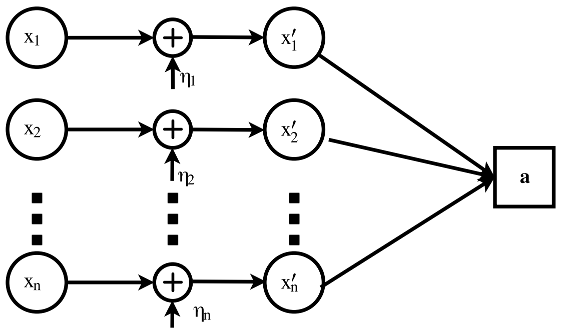

Figure 3.1: Noise model for linear regression.

vector $\mathbf{x}$ is denoted by $\mathbf{x}'$. The learning algorithm receives noisy versions of each feature. A Gaussian random variable η_j with mean zero and variance σ_j^2 is added to the jth feature. We use Gaussian noise, as the output of low power sensors used in wireless sensor networks is often corrupted by Gaussian noise when it arrives at the fusion center. Depending on the quality of the sensor, different sensors may have different levels of noise. Therefore our model is able to handle different levels of noise for different features.

3.1 Replication

Next we discuss the effects of channel coding on the noise applied to each feature. As we previously described, we rely on replication for two reasons: first, to deal with the fact that the encoder may not have access to the noiseless, true data, and, secondly, because replication codes enable us to provide an appropriate level of protection to each feature. Replication involves repeating the jth feature k_j times, i.e., the codeword $c_j = (x_j, x_j, \ldots, x_j)$. The replicated values are passed through the noisy channel and are corrupted. Depending on the type of noise model, replication coding mitigates the probability of feature error in different ways. For our Gaussian noise model, the decoding process involves taking the average of the noisy versions of the repetitions of a feature: $\frac{1}{k_j}(\sum_{i=1}^{k_j} c_j(i))$. This effectively reduces the overall variance of the noise ${\sigma_j}^2$ of the jth feature:

$$\begin{aligned} Var\left[\frac{1}{k_j}\sum_{i=1}^{k_j}(c_j(i)+\eta_j(i))\right] &= \frac{1}{{k_j}^2} Var\left[\sum_{i=1}^{k_j}\eta_j(i)\right] \\ &= \frac{1}{{k_j}^2}(k_j{\sigma_j}^2) = \frac{{\sigma_j}^2}{k_j}. \end{aligned}$$

Thus replication reduces the variance of the noise added to the feature by a factor equal to the number of repetitions.

Now that we have defined how replication reduces the effect of noise on the data, we discuss how to allocate redundancy given a budget of N replications in order to reduce the effects of noise on the predictions made by the algorithm in an informed way.

3.2 Expected Noise Loss

We measure the expected loss due to noise (between the output of the noisy algorithm and the noiseless algorithm) with a mean-squared error (MSE) approach. The MSE is the following:

$$\begin{aligned}
&\mathbb{E}\left[\mathbf{a}^T\mathbf{x} - \mathbf{a}^T\mathbf{x}'\right] \\
&= \mathbb{E}[(\mathbf{a}^T(\mathbf{x} - \mathbf{x}'))^T(\mathbf{a}^T(\mathbf{x} - \mathbf{x}'))] \\
&= \mathbb{E}[((\mathbf{x} - \mathbf{x}')^T\mathbf{a}\mathbf{a}^T(\mathbf{x} - \mathbf{x}'))] \\
&= Tr(\mathbf{a}\mathbf{a}^T Cov(\mathbf{x} - \mathbf{x}')) + \mathbb{E}[\mathbf{x} - \mathbf{x}']^T\mathbf{a}\mathbf{a}^T\mathbb{E}[\mathbf{x} - \mathbf{x}'].
\end{aligned} \tag{3.1}$$

The final step uses the identity that for any matrix $\mathbf{A}$ and vector $\mathbf{c}$ we have

$$\mathbb{E}[\mathbf{c}^T\mathbf{A}\mathbf{c}] = Tr(\mathbf{A}Cov(\mathbf{c})) + \mathbb{E}[\mathbf{c}]^T\mathbf{A}\mathbb{E}[\mathbf{c}].$$

We note that using the MSE (as opposed to MAE) is motivated in part by the fact that linear regression involves minimizing square error itself.

We use an example to illustrate why optimizing the redundancy assignment is necessary to improve the performance of the regression model. The example helps build intuition regarding the technique that can be used to improve the performance of the algorithm.

Example: Consider a small data set generated from an underlying system where the model is $x = 0.5y + 16z$, where y and z are the measured features which have noise added to them. When the features are corrupted equally by zero mean Gaussian noise with a variance of 4, then the mean squared loss at the output of the algorithm is 1025. With uniform repetition coding of 3 replications for each feature, the loss is reduced to 341.7. However by exhaustive search we can find that the optimal redundancy allocation is 1 units to y and 5 units to feature z. This yields a mean squared loss of at the output 205.8. This simple example motivates the need for a strategy to efficiently optimize coding (that is, redundancy allocations).

3.2.1 Optimizing the MSE loss

Next we analyze the continuous version of the optimization problem. Using the same idea as the expression in (3.1), we derive the expected loss for the current scenario. The combined noise vector η is a zero mean Gaussian random vector, with covariance matrix $\mathbf{\Sigma} = diag(\sigma_1^2, \ldots, \sigma_n^2)$. The expected loss is given by

$$\begin{aligned}
&\mathbb{E}\left[\mathbf{a}^T\mathbf{x} - \mathbf{a}^T\mathbf{x}'\right] \\
&= Tr(\mathbf{a}\mathbf{a}^T Cov(\mathbf{x} - \mathbf{x}')) + \mathbb{E}[\mathbf{x} - \mathbf{x}']^T \mathbf{a}\mathbf{a}^T \mathbb{E}[\mathbf{x} - \mathbf{x}'] \\
&= \sum_{j=1}^{n} a_j^2 {\sigma_j}^2.
\end{aligned}$$

We are now ready to introduce the key optimization for our informed replication approach. We have a budget of N repetitions and we wish to allocate these among features in such a way that our resulting expected loss is minimized. We define the redundancy allocation vector $\mathbf{k} = (k_1, ..., k_n)$, where the ith feature is replicated k_i times for $1 \leq i \leq n$. We set up the constrained optimization problem as follows:

$$\min_{k_1,\ldots,k_n} g(\mathbf{k}) = \sum_{j=1}^{n} \frac{a_j^2 {\sigma_j}^2}{k_j},$$

subject to

$$\sum_{j=1}^{n} k_j = N.$$

where k_j is a positive integer.

Note that due to the integer constraint, the problem is not convex. However, if we relax the constraint and allow k_j to be real valued, the optimization problem becomes convex. We solve this optimization problem using the method of Lagrange multipliers.

Theorem 1. *The optimal redundancy allocation for the continuous relaxation of the problem for the jth feature is given by*

$$k_j = \frac{|a_j \sigma_j|}{\sum_{j=1}^{n} |a_j \sigma_j|} N.$$

Proof. Define the function g' and as follows:

$$g'(\mathbf{a}, \mathbf{k}, \lambda) = \sum_{j=1}^{n} \frac{a_j^2 {\sigma_j}^2}{k_j} - \lambda(\sum_{j=1}^{n} k_j - N).$$

Taking the partial derivatives and setting them to zero gives the following relationship between λ and k_j

$$k_j = |a_j \sigma_j| \sqrt{\frac{-1}{\lambda}}.$$

Substituting this back into the sum constraint gives the following solution for λ:

$$\lambda = \frac{-(\sum_{j=1}^{n} |a_j \sigma_j|)^2}{N^2}.$$

We then solve for k_j by substituting the value of λ in the equation for k_j

$$\begin{aligned} k_j &= |a_j\sigma_j|\sqrt{\frac{-N^2}{-(\sum_{j=1}^{n}|a_j\sigma_j|)^2}} \\ &= \frac{|a_j\sigma_j|}{\sum_{j=1}^{n}|a_j\sigma_j|}N. \end{aligned}$$

□

Note that if this solution yields integer values for all k_j's, then this is the optimal solution for the integer version of the problem as well. However, if any of obtained values are not integers, then we do the following. Use k_j for all values of j. There are at most $n-1$ units left after this for allocation. We can try all possible allocations for these bits to find the solution to the integer version of the problem.

3.2.2 Solution using submodular optimization

While the continuous relaxation gives an efficient technique to approach the optimal solution for our optimization problem, it still requires an exhaustive search to allocate up to $n-1$ of the final redundancy units. To solve this problem efficiently, we must take the integer constraints into consideration. As such, we use submodular optimization. Submodular optimization is a technique used to solve optimization problems that involve integer constraints [40]. It is a computationally efficient and nearly optimal technique, as shown in [40]. The algorithm, which is a greedy technique, involves allocating units of redundancy one at a time iteratively. Given m available units of redundancy, consider the redundancy allocation vector $\mathbf{k}(i) = (k_1(i), \ldots, k_n(i)), 1 \leq i \leq m$ which denotes the units of redundancy allocated to each vector at the ith step of the algorithm. Then at step $i+1$, we assign an additional unit of redundancy to the lth feature. This gives the vector $\mathbf{k}(i+1) = (k_1(i), \ldots, k_l(i)+1, \ldots, k_n(i))$. The goal of the minimization at each step is to find the $\min_l \sum_{j=1}^{n} \frac{a_j^2 {\sigma_j}^2}{k_j}, 1 \leq l \leq n$. This process continues until all units of redundancy are allocated.

We show that our optimization function is submodular. The objective function must fulfill the conditions of monotonicity and diminishing returns in order to validate the use of submodular optimization. The monotonicity statement is as follows:

Lemma 3. *Given redundancy allocation vectors* $\mathbf{k} = (k_1, \ldots, k_i, \ldots, k_n)$ *and* $\tilde{\mathbf{k}} = (k_1, \ldots, k_{i-1}, \tilde{k}_i, k_{i+1}, \ldots, k_n)$*, where* $k_i > \tilde{k}_i$*. Then* $f(\mathbf{k}) < f(\tilde{\mathbf{k}})$*.*

The proof for Lemma 3 is immediate, as the optimization function is inversely proportional to k_i for all i. Lemma 3 shows that the cost function decreases as more units of redundancy are allocated. Next we consider the diminishing returns property. For this property to hold, repeated allocation of redundancy units to the same feature must yield reduced improvements.

Lemma 4. *Consider redundancy allocation vectors* $\mathbf{k} = (k_1, \ldots, k_n)$ *and* $\mathbf{z} = (z_1, \ldots, z_n)$*, where* $k_j \leq z_j$ *for all* j*. Then for all* i *,*

$$\sum_j a_j^2\sigma_j^2/k_j - (\sum_{j\neq i} a_j^2\sigma_j^2/k_j + a_i^2\sigma_i^2/(k_i+1)) \geq$$
$$\sum_j a_j^2\sigma_j^2/z_j - (\sum_{j\neq i} a_j^2\sigma_j^2/z_j + a_i^2\sigma_i^2/(z_i+1)).$$

Proof. We prove the lemma by expanding the summation on both sides and canceling out terms accordingly. This simplifies the expression to

$$a_i^2\sigma_i^2/k_i - a_i^2\sigma_i^2/(k_i+1) \geq a_i^2\sigma_i^2/z_i - a_i^2\sigma_i^2/(z_i+1)$$
$$\Rightarrow 1/k_i - 1/(k_i+1) \geq 1/z_i - 1/(z_i+1)).$$

This final statement holds true when $k_i \leq z_i$ and k_i, z_i are integers, which is the indeed the case. □

Thus the cost function is indeed submodular. We use the floor of the redundancy allocations obtained from the continuous relaxation and then allocate the remaining $n-1$ units using submodular optimization.

3.3 Optimizing the linear regression model

So far we have worked on optimizing the redundancy allocation given a fixed set of regression coefficients. This reflects the assumption that we have a fixed model, without the ability to change it. On the other hand, as long as we know the test data noise characteristics, we can *optimize* the model in order to minimize the joint square error (from regression and noise simultaneously).

Motivated by the previous notion, in this section we show how we optimize the model for regression given the regression data and the noise model. Afterwards, we further select the optimal replication scheme. We assume that the noise on each feature is independent of noise on all other features. Let $\mathbf{X} \in \mathbb{R}^{p\times n}$ be the matrix of p data points with n features. Let the matrix $\mathbf{X}' = \mathbf{X} + \mathbf{N}$ be the noisy data set where $\mathbf{N} \in \mathbb{R}^{p\times n}$ is a matrix where each row is an independent realization of an n-variate Gaussian random vector with zero mean and $\mathbf{\Sigma}$ as the covariance matrix. Assume that each element of this random vector is independent and as such, the covariance matrix is a diagonal matrix, i.e. $\mathbf{\Sigma} = diag(\sigma_1^2, \ldots, \sigma_n^2)$. Then $E[\mathbf{N}] = \mathbf{0}$ and $E[\mathbf{N}^T\mathbf{N}] = p\mathbf{\Sigma}$ (by the expectation of the Wishart distribution[41]). The objective function to minimize in this case is as follows:

$$
\begin{aligned}
&\min_a \mathbb{E}[\mathbf{y} - \mathbf{X}'\mathbf{a}^2] \\
&\quad = \min_{\mathbf{a}} \mathbb{E}[(\mathbf{y} - (\mathbf{X}+\mathbf{N})\mathbf{a})^T(\mathbf{y} - (\mathbf{X}+\mathbf{N})\mathbf{a})] \\
&\quad = \min_{\mathbf{a}} \mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbb{E}[(\mathbf{X}+\mathbf{N})]\mathbf{a} \\
&\quad - \mathbf{a}^T\mathbb{E}[(\mathbf{X}+\mathbf{N})^T]\mathbf{y} + \mathbf{a}^T\mathbb{E}[(\mathbf{X}+\mathbf{N})^T(\mathbf{X}+\mathbf{N})]\mathbf{a} \\
&\quad = \min_{\mathbf{a}} \mathbf{y} - \mathbf{X}\mathbf{a}^2 + p\mathbf{a}^T\Sigma\mathbf{a}.
\end{aligned}
$$

The problem takes the form of the regularized least squares problem. The solution to the above regression problem is as follows:

$$\mathbf{a} = (\mathbf{X}^T\mathbf{X} + p\mathbf{\Sigma})^{-1}\mathbf{X}^T\mathbf{y}.$$

Now we plug this value of **a** back into the objective function and attempt to optimize over the elements of the matrix $\mathbf{\Sigma}$. While we have reduced the terms that involve components of $\mathbf{\Sigma}$ for the optimization, we must still approximate the inverse of $(\mathbf{X}^T\mathbf{X} + p\mathbf{\Sigma})$ if we wish to find a solution to the optimization problem for the noise variance. Plugging in the value for the optimal **a** and simplifying gives the following cost function to minimize:

$$\begin{aligned}&\min_{\sigma_1,\ldots,\sigma_n} \mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}(\mathbf{X}^T\mathbf{X} + p\mathbf{\Sigma})^{-1}\mathbf{X}^T\mathbf{y} \\ &\qquad = \min_{\sigma_1,\ldots,\sigma_n} \mathbf{y}^T(\mathbf{I} - \mathbf{X}(\mathbf{X}^T\mathbf{X} + p\mathbf{\Sigma})^{-1}\mathbf{X}^T)\mathbf{y}.\end{aligned}$$

Note that the objective function expression cannot be negative as it is derived from the linear regression cost, which is a nonnegative value. Thus, the solution to the above optimization problem is also the solution of the following problem:

$$\max_{\sigma_1,\ldots,\sigma_n} \mathbf{y}^T\mathbf{X}(\mathbf{X}^T\mathbf{X} + p\mathbf{\Sigma})^{-1}\mathbf{X}^T\mathbf{y}.$$

3.3.1 Solution by submodular optimization

Once again, we attempt to solve the optimization problem using submodular optimization. Before we begin we state several facts that will be used to prove that the submodular optimization approach is applicable.

Fact 3. *[42] Let the matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ be symmetric with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$. Then for all $\mathbf{x} \in \mathbb{R}^n$ $\lambda_n\mathbf{x}^T\mathbf{x} \leq \mathbf{x}^T\mathbf{A}\mathbf{x} \leq \lambda_1\mathbf{x}^T\mathbf{x}$.*

This fact allows us to bounds the quadratic form involving a matrix with its eigenvalues.

We next look at the bounds on the eigenvalues for a matrix that has been perturbed.

Fact 4. *[43] Let $\mathbf{A}, \mathbf{A}+\mathbf{E} \in \mathbb{R}^{n\times n}$ be symmetric matrices and $\lambda_k(\mathbf{G})$ denote the kth largest eigenvalue of the matrix $\mathbf{G}$. Then $\lambda_n(\mathbf{E}) + \lambda_k(\mathbf{A}) \leq \lambda_k(\mathbf{A}+\mathbf{E}) \leq \lambda_1(\mathbf{E}) + \lambda_k(\mathbf{A})$.*

Finally we also state a fact that relates the eigenvalues of a matrix with its inverse.

Fact 5. *[42] Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be a symmetric non-singular matrix. Let $\lambda_1 \geq \ldots \geq \lambda_n$ be the eigenvalues of $\mathbf{A}$. Then the eigenvalues of $\mathbf{A}^{-1}$ are $\frac{1}{\lambda_n} \geq \ldots \geq \frac{1}{\lambda_1}$.*

Now we are ready to prove the submodularity properties for the joint optimization. We begin by stating the lemma required to show that successive allocations of redundancy units causes the objective function to decrease.

Lemma 5. *Given two redundancy allocations defined by the covariance matrices $\mathbf{\Sigma} = p \times diag(\frac{\sigma_1^2}{k_1}, \ldots, \frac{\sigma_i^2}{k_i}, \ldots, \frac{\sigma_n^2}{k_n})$ and $\tilde{\mathbf{\Sigma}} = p \times diag(\frac{\sigma_1^2}{k_1}, \ldots, \frac{\sigma_n^2}{k_{i-1}}, \frac{\sigma_i^2}{\tilde{k_i}}, \frac{\sigma_n^2}{k_{i+1}}, \ldots, \frac{\sigma_n^2}{k_n})$ respectively, where $\tilde{k_i} < k_i$ for a particular i where $1 \leq i \leq n$. We define $h(\mathbf{\Sigma}) = \mathbf{y}^T(\mathbf{I} - \mathbf{X}(\mathbf{X}^T\mathbf{X}+\mathbf{\Sigma})^{-1}\mathbf{X}^T)\mathbf{y}$. Then $h(\mathbf{\Sigma}) \leq h(\tilde{\mathbf{\Sigma}})$.*

Proof. Firstly $\mathbf{y}^T\mathbf{y}$ is common to both $h(\mathbf{\Sigma})$ and $h(\tilde{\mathbf{\Sigma}})$. Next, we use the above defined facts to put bounds on the eigenvalues for inverse of a perturbed matrix.

$$\begin{aligned}\frac{1}{\lambda_{n-l+1}(\mathbf{X}^T\mathbf{X}) + \lambda_1(\mathbf{\Sigma})} &\leq \lambda_l((\mathbf{X}^T\mathbf{X} - \mathbf{\Sigma})^{-1}) \\ &\leq \frac{1}{\lambda_{n-l+1}(\mathbf{X}^T\mathbf{X}) + \lambda_n(\mathbf{\Sigma})}.\end{aligned}$$

Note that the indices are reversed due to the fact that we are defining the bounds of the eigenvalue of a matrix using the eigenvalues of its inverse. We define $\mathbf{u} = \mathbf{X}^T\mathbf{y}$. Then $\mathbf{u}^T(\mathbf{X}^T\mathbf{X} - \mathbf{\Sigma})^{-1}\mathbf{u} \leq \frac{1}{\lambda_n(\mathbf{X}^T\mathbf{X})+\lambda_n(\mathbf{\Sigma})}\mathbf{u}^T\mathbf{u}$. Similarly, for $\tilde{\mathbf{\Sigma}}$ we have $\mathbf{u}^T(\mathbf{X}^T\mathbf{X} - \tilde{\mathbf{\Sigma}})^{-1}\mathbf{u} \leq \frac{1}{\lambda_n(\mathbf{X}^T\mathbf{X})+\lambda_n(\tilde{\mathbf{\Sigma}})}\mathbf{u}^T\mathbf{u}$. As the covariance matrices are diagonal, we have the fact that $\lambda_n(\mathbf{\Sigma}) \leq \lambda_n(\tilde{\mathbf{\Sigma}})$. Therefore the quadratic form involving $\mathbf{\Sigma}$ is greater than or equal to the quadratic form involving $\tilde{\mathbf{\Sigma}}$. As such $h(\mathbf{\Sigma}) \leq h(\tilde{\mathbf{\Sigma}})$. $\square$

Next we wish to prove the diminishing returns property for this optimization problem. The diminishing returns property is as follows:

Lemma 6. *Given redundancy allocation vectors* $\mathbf{k} = (k_1, \ldots, k_n)$ *and* $\mathbf{z} = (z_1, \ldots, z_n)$, *where* $k_j \leq z_j$ *for all* j *where* $1 \leq j \leq n$. *These define two covariance matrices* $\mathbf{\Sigma}_k = p \times diag(\sigma_1^2/k_1, \ldots, \sigma_n^2/k_n)$ *and* $\mathbf{\Sigma}_z = p \times diag(\sigma_1^2/z_1, \ldots, \sigma_n^2/z_n)$. *We further define* $\mathbf{\Sigma}_{k+1} = p \times diag(\sigma_1^2/k_1, \ldots, \sigma_i^2/(k_i{+}1), \ldots, \sigma_n^2/k_n)$ *and* $\mathbf{\Sigma}_{z+1} = p \times diag(\sigma_1^2/z_1, \ldots, \sigma_i^2/(z_i{+}1), \ldots, \sigma_n^2/z_n)$. *Then for all* i $h(\mathbf{\Sigma_k}) - h(\mathbf{\Sigma_{k+1}}) \geq h(\mathbf{\Sigma_z}) - h(\mathbf{\Sigma_{z+1}})$, *where* $1 \leq i \leq n$.

Proof. Using the bounds established for the monotonicity proof, we attempt to show the diminishing returns property. Firstly we note that $\lambda_n(\mathbf{\Sigma_k}) - \lambda_n(\mathbf{\Sigma_{k+1}}) \leq \sigma_i^2/(k_i) - \sigma_i^2/(k_i{+}1)$. Also as $k_j \leq z_j$ for all j (where $1 \leq j \leq n$) and are integers, we have for all i (where $1 \leq i \leq n$):

$$\sigma_i^2/(k_i) - \sigma_i^2/(k_i + 1) \geq \sigma_i^2/(z_i) - \sigma_i^2/(z_i + 1).$$

Next we expand the following expression

$$\begin{aligned}
&h(\mathbf{\Sigma_k}) - h(\mathbf{\Sigma_{k+1}}) \\
&= \mathbf{y}^T\mathbf{X}(\mathbf{X}^T\mathbf{X} + \mathbf{\Sigma_{k+1}})^{-1}\mathbf{X}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}(\mathbf{X}^T\mathbf{X} + \mathbf{\Sigma_k})^{-1}\mathbf{X}^T\mathbf{y}.
\end{aligned}$$

We can use the upper bound for the first term and the lower bound for the second term to get

$$\begin{aligned}
&h(\mathbf{\Sigma_k}) - h(\mathbf{\Sigma_{k+1}}) \\
&\leq \left(\frac{1}{\lambda_n(\mathbf{X}^T\mathbf{X}) + \lambda_n(\mathbf{\Sigma_{k+1}})} - \frac{1}{\lambda_1(\mathbf{X}^T\mathbf{X}) + \lambda_1(\mathbf{\Sigma_k})}\right)\mathbf{y}^T\mathbf{X}\mathbf{X}^T\mathbf{y}.
\end{aligned}$$

Similarly for $\mathbf{\Sigma_z}$ we have

$$\begin{aligned}
&h(\mathbf{\Sigma_z}) - h(\mathbf{\Sigma_{z+1}}) \\
&\leq \left(\frac{1}{\lambda_n(\mathbf{X}^T\mathbf{X}) + \lambda_n(\mathbf{\Sigma_{z+1}})} - \frac{1}{\lambda_1(\mathbf{X}^T\mathbf{X}) + \lambda_1(\mathbf{\Sigma_z})} \right) \mathbf{y}^T\mathbf{X}\mathbf{X}^T\mathbf{y}.
\end{aligned}$$

We note that:

$$\left(\frac{1}{\lambda_n(\mathbf{X}^T\mathbf{X}) + \lambda_n(\mathbf{\Sigma_{k+1}})} - \frac{1}{\lambda_n(\mathbf{X}^T\mathbf{X}) + \lambda_n(\mathbf{\Sigma_{z+1}})} \right) \mathbf{y}^T\mathbf{X}\mathbf{X}^T\mathbf{y}$$

$$\geq \left(\frac{1}{\lambda_1(\mathbf{X}^T\mathbf{X}) + \lambda_1(\mathbf{\Sigma_k})} - \frac{1}{\lambda_1(\mathbf{X}^T\mathbf{X}) + \lambda_1(\mathbf{\Sigma_z})} \right) \mathbf{y}^T\mathbf{X}\mathbf{X}^T\mathbf{y}.$$

The above inequality is true due to the fact that the terms on the right hand side involve inverses of larger terms than that on the left hand side. We can rearrange the inequality to bring it into the desired form. In addition it is possible to show that the upper bounds on both sides of the inequality will be equally tight due to the fact that both sides represent costs within one additional unit of each other. Therefore we have:

$$\left(\frac{1}{\lambda_n(\mathbf{X}^T\mathbf{X}) + \lambda_n(\mathbf{\Sigma_{k+1}})} - \frac{1}{\lambda_1(\mathbf{X}^T\mathbf{X}) + \lambda_1(\mathbf{\Sigma_k})} \right) \mathbf{y}^T\mathbf{X}\mathbf{X}^T\mathbf{y}$$

$$\begin{aligned}
&\geq \left(\frac{1}{\lambda_n(\mathbf{X}^T\mathbf{X}) + \lambda_n(\mathbf{\Sigma_{z+1}})} - \frac{1}{\lambda_1(\mathbf{X}^T\mathbf{X}) + \lambda_1(\mathbf{\Sigma_z})} \right) \mathbf{y}^T\mathbf{X}\mathbf{X}^T\mathbf{y}, \\
&\Rightarrow h(\mathbf{\Sigma_k}) - h(\mathbf{\Sigma_{k+1}}) \geq h(\mathbf{\Sigma_z}) - h(\mathbf{\Sigma_{z+1}}).
\end{aligned}$$

This concludes the proof for submodularity for the joint optimization. $\square$

Thus, given a budget of N repetition, we can use submodular optimization to efficiently allocate the redundancy optimally. We can now revisit the original example in Section I and apply this technique. The example is as follows:

Example: Let us assume that the data is generated using the model ($x = 0.5y + 16z$). However, now the redundancy allocation must be done during the training phase. Given the fact that the noise variance vector is know to be $[1, 20]$ (very high noise on feature 2), with the same redundancy budget of 6 units, we find that the optimal redundancy allocation remains the same. However, the regression weights that are calculated change to 0.53 for feature y and 15.9 for feature z. We will see more examples of changes in regression weights in the experimental results section.

3.4 Experimental Results

In this section we present experimental results which demonstrate the advantage our optimization methods provide (over an uninformed coding scheme). We first present results for allocating redundancy when the regression coefficients are fixed. We use the following regression model: $u = 3v + 9x + 7y + 4z$. The vector of variances is $[2, 2, 4, 4]$. For a redundancy budget of 40 units, the optimal redundancy allocation is $k_v = 4, k_x = 13, k_y = 15, k_z = 8$ units. The mean squared error for the optimal allocation is 38.03 and for uniform allocation, it is 44.

Next we look at how the cost varies as redundancy allocation increases from 8 units upwards. We plot the mean squared cost for the optimal allocation and the uniform allocation for the above setup. The plot is shown in Figure 3.4. We can see that the optimal allocation provides a clear benefit over the uniform allocation in terms of the MSE. Furthermore we see that benefit is present even at high values of redundancy. However there is the factor of diminishing returns which indicates that for a high enough redundancy budget, the optimal allocation would be the uniform allocation.

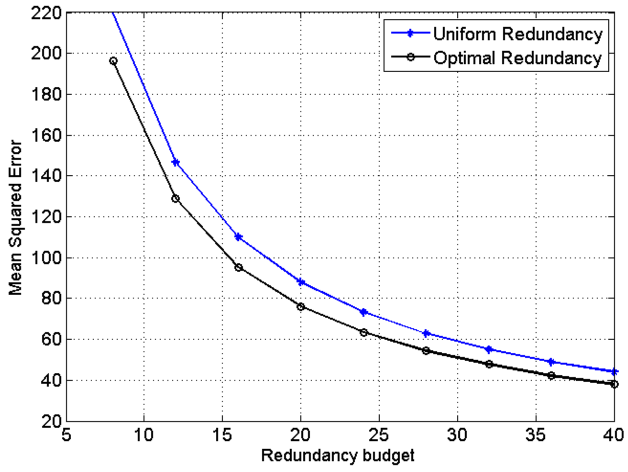

Figure 3.2: MSE for Optimal Allocation vs Uniform Allocation for the 1st set of regression coefficients.

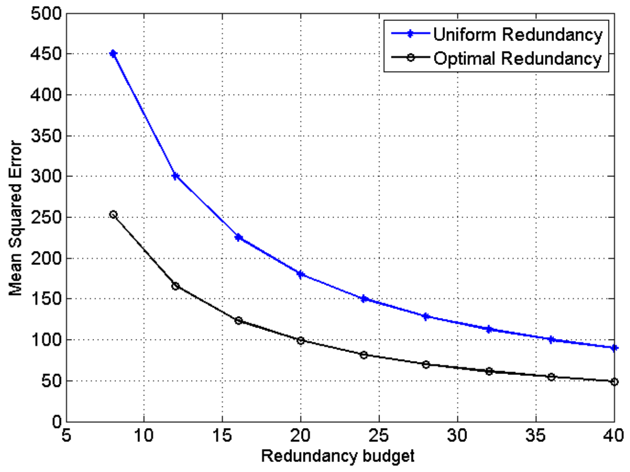

Figure 3.3: MSE for Optimal Allocation vs Uniform Allocation for the 2nd set of regression coefficients.

To further see the benefits of our allocation strategy, we use the following example: Let the regression model be $u = v + 9x + 7y + 4z$, with the noise variance vector as $[10, 10, 1, 2]$. This means that the noise in features 1 and 2 is very high, and the noise in features 3 and 4 is comparatively lower. For this case, the plot for the optimal MSE and uniform MSE over a range of values of the redundancy is shown in Figure 3.3.

The advantage of our allocation scheme is very clear in this scenario. The optimal allocation scheme produces a MSE which is roughly half of the MSE for the uniform allocation for almost all values of the redundancy budget. As our allocation scheme takes into account

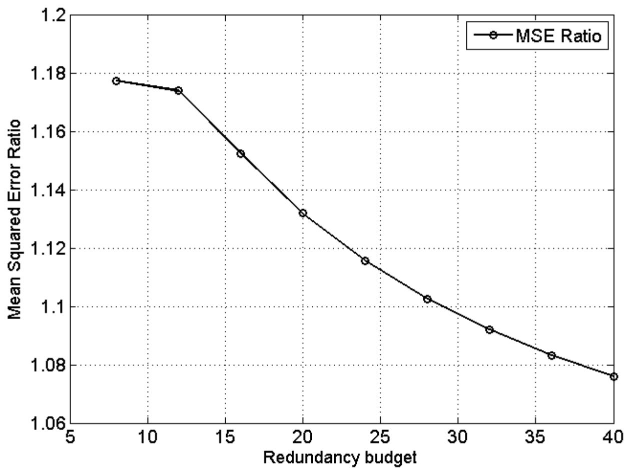

Figure 3.4: Ratio of uniform MSE to optimal MSE for the power data set.

both the regression weights and noise variance for each feature, it is able to provide a consistent benefit over the uniform scheme in this case. These two examples also illustrate how the amount of improvement in the model's performance depends on the noise distribution and model parameters.

Next we consider the case where the optimal allocation must be made during the training phase. We use the power production dataset [44]. The dataset contains 9568 data points collected from a power plant working at full load. It contains four features which are: hourly average ambient variables Temperature (T), Ambient Pressure (AP), Relative Humidity (RH) and Exhaust Vacuum (V). The target variable is the net hourly output of the power plant. To start with, we consider the following noise variance vector $[8, 8, 1, 1]$. This means that the first two features are affected by noise with a variance of 8 units, while the remaining features are affected by noise with variance 1. We compute the optimal allocation of redundancy units when only 8 units of redundancy are available. The resulting redundancy allocation is $k_T = 5, k_{AP} = 1, k_{RH} = 1$, and $k_V = 1$. We can see that despite the fact that the first and second features are corrupted by noise with high variance, only the first feature gets the majority of the redundancy budget. The output MSE for this allocation is 27, while the output MSE for uniform allocation is 32.2, thus the optimal allocation provides about 20% improvement over the uniform case. We find that the regression weight vector (ignoring the intercept) $\mathbf{a}$ obtained using the optimal allocation is $[-1.7761, -0.2888, 0.1338, -0.1179]$. On the other hand, the noiseless training gives us the following vector of regression coefficients: $[-1.9689, -0.2316, 0.0772, -0.1620]$, showing that it is necessary to change the regression coefficients when the data is expected to be noisy.

Next we plot the ratio of uniform allocation MSE to optimal allocation MSE for a range of redundancy allocations for this setup. The plot is shown in Figure 3.4. We can see in the figure that using the submodular optimization technique gives a significant advantage over the uniform allocation for low values of redundancy budget. This advantage decreases as the budget increases, however it does not completely vanish. In scenarios where highly accurate predictions are required, even the slightest increase in accuracy is valuable. Our redundancy allocation technique ensures that the model is as accurate as possible with the given redundancy budget.

CHAPTER 4

Conclusion

In this section, we presenting concluding remarks for the techniques presented in this thesis. First we discuss the results seen in chapter 2. We studied a noisy LDPC decoding problem which was different from the conventional model, where the messages passed between the variable and check nodes are corrupted by a binary symmetric channel. In the problem presented in this thesis, we developed a model inspired by the noise induced on decoder due to radiation events. This model allows for a varying fraction of variable nodes to enter a noisy state with errors and erasure. The fraction varies according to a Poisson process based on the M/M/∞ queue model. We performed density evolution analysis, which is an excellent technique to predict the performance of a code ensemble. We used this analysis to gain insights regarding the effects of different radiation parameters(i.e. nodes affected per event, event arrival rate and duration) on the decoder output in terms of the BER. We found that there were two regimes, one where the number of nodes affected per event had the dominant effect on the decoder performance and the other where the memory error probability had the dominant effect. We studied the performance of the decoder as a function of the voting threshold for the variable nodes and found how increasing the threshold at higher levels of radiation improves decoder performance. We found that the traditional i.i.d model gives a pessimistic estimate of the decoder's performance when compared to the queue based model that we introduce. We presented results from finite length experiments to support the conclusions drawn from the theoretical analysis. This work has been presented initially in the IEEE Asilomar Conference on Signals, Systems, and Computers in 2016. A more comprehensive analysis is presented in the IEEE Transactions on Communication in 2017 [45].

Next we discuss the results seen in chapter 3. In chapter 3, we considered informed replication techniques to protect linear regression test data from noise. We presented a strategy to compute the optimal redundancy allocation (in terms of the divergence from the hypothetical noise-free case on the output) for a regression model when applied to data that has been corrupted by Gaussian noise. We showed how this allocation of redundancy is a function of the regression coefficients and noise variance for each feature when the regression coefficients are given. We further presented a technique to obtain the optimal redundancy allocation for noisy features during the training stage. We showed how the regression coefficients become a function of the redundancy allocation while the model is being trained, along with the improvement of our redundancy allocation techniques compared to a uniform allocation. The optimal redundancy allocation is a step towards identifying levels of protection for each feature. Future work can involve application of these techniques to different noise models such as the binary symmetric channel and using this concept of protection to build more complex codes for regression. This work has been accepted at 55th Annual Allerton Conference on Communication, Control, and Computing in 2017.

In this thesis, we have seen two interesting problems where corrupted data impacts the performance of the algorithm which uses the data. We have seen how taking leverage of the additional information provided(i.e. changing the noise model or using the regression weights) allows us to improve the performance of the algorithm by allocating redundancy optimally(for linear regression) or by changing the decoding parameters based on the noise level(for the LDPC decoder). On a high-level, we have found that when dealing with algorithms operating on noisy data, it is beneficial to choose codes based on metrics defined using the algorithm performance and not just traditional metrics such as Hamming distance.

References

[1] Mars Science Laboratory (MSL) mast camera (mastcam) instrument description, `http://www.msss.com/science/msl-mastcam-instrument-description.php`, 2012.

[2] Memory reformat planned for Opportunity Mars rover, `http://www.jpl.nasa.gov/news/news.php?feature=4275`, 2014.

[3] F. Sala, C. Schoeny, D. Divsalar, and L. Dolecek, "Asymmetric error-correcting codes for Flash memories in high-radiation environments," in *Proc. IEEE ISIT*, Hong Kong, Jun. 2015.

[4] F. Sala, C. Schoeny, D. Divsalar, and L. Dolecek, "Asymmetric ECCs for Flash in high radiation environments," in *Proc. IEEE Asilomar Conf. on Signals, Systems, and Computers*, Pacific Grove, CA, Nov. 2015.

[5] Y. Li, D. Sheldon, A. Ramos, and J. Bruck, "Error characterization and mitigation for 16nm MLC NAND flash memory under total ionizing dose effect," in *Proc. IEEE Int. Reliability Physics Symp. (IRPS)*, Apr., 2016.

[6] Y. Cassuto *et al.*, "Codes for asymmetric limited-magnitude errors with application to multi-level flash memories," *IEEE Trans. Inf. Theory*, vol. 56, no. 4, pp. 1582-1596, Apr. 2010.

[7] R. Gabrys, E. Yaakobi, and L. Dolecek,"Graded bit-error-correcting codes with applications to flash memory," *IEEE Trans. Inf. Theory*, vol. 59, no. 4, pp. 2315-2327, Apr. 2013.

[8] C.-H. Huang and Y. Li and L. Dolecek, "Belief propagation algorithms on noisy hardware," *IEEE Trans. Commun.*, vol. 63, no. 1, pp. 11-24, Jan. 2015.

[9] C.-H. Huang and Y. Li and L. Dolecek, "Gallager B LDPC decoder with transient and permanent errors," *IEEE Trans. Commun.*, vol. 62, no. 1, pp. 15-28, Jan. 2014.

[10] S. M. S. Tabatabaei, H. Cho and L. Dolecek, "Gallager B decoder on noisy hardware," *IEEE Trans. Commun*, vol. 61, no. 5, pp. 1660-1673, May 2013.

[11] L. Varshney, "Performance of LDPC codes under faulty iterative decoding," *IEEE Trans. Info. Theory*, vol. 57, no. 7, pp. 4427-4444, Jul. 2011.

[12] F. Leduc-Primeau and W. J. Gross, "Faulty Gallager-B decoding with optimal message repetition," in *Proc. IEEE Allerton Conference on Communication, Control, and Computing*, Monticello, IL, Oct. 2012.

[13] A. Balatsoukas-Stimming and A. Burg, "Density evolution for minsum decoding of LDPC codes under unreliable message storage," *IEEE Comm. Letters*, vol. 18, no. 5, pp. 849-852, May. 2014.

[14] S. Brkic, O. Al Rasheed, P. Ivanis, and B. Vasic, "On fault tolerance of the Gallager B decoder under data-dependent gate failures," *IEEE Comm. Letters*, vol. 19, no. 8, pp. 1299-1302, Aug. 2015.

[15] S. Brkic, P. Ivanis, and B. V. Vasic, "Reliability of memories built from unreliable components under data-dependent gate failures," *IEEE Comm. Letters*, vol. 19 no. 12, pp. 2098-2101, Dec. 2015.

[16] F. Leduc-Primeau, F. R. Kschischang, and W. J. Gross, "Energy optimization of LDPC decoder circuits with timing violations," in *Proc. IEEE Int. Conf. on Comm. (ICC)*, London, UK, Jun. 2015.

[17] P. Schalfer, C. Huang, C. Schoeny, C. Weis, Y. Li, N. Wehn, and L. Dolecek, “Error Resilience and Energy Efficiency: an LDPC Decoder Study," in Proc. DATE, Dresden, Germany, Mar. 2016.

[18] J. Mu *et al.*, “The impact of faulty memory bit cells on the decoding of spatially-coupled LDPC codes,” in *Proc. IEEE Asilomar Conf. on Signals, Systems, and Computers.*, Pacific Grove, CA, Nov. 2015, pp. 1627-1631.

[19] S. Gerardin and A. Paccagnella, “Present and future non-volatile memories for space,” *IEEE Trans. Nuclear Science*, vol. 57, no. 6, pp. 3016-3039, Dec. 2010.

[20] S. Gerardin *et al.*, “Radiation effects in flash memories,” *IEEE Trans. Nuclear Science*, vol. 60, no. 3, pp. 1953-1969, Jun. 2013.

[21] F. Irom et al., “Catastrophic Failure in Highly Scaled Commercial NAND Flash Memories,” *IEEE Trans. Nuclear Science*, vol. 57, no. 1, pp. 266-271, Feb. 2010.

[22] V. G. Kulkarni, *Modeling and analysis of stochastic systems*, first ed.: Chapman & Hall.

[23] M.P. Fossorier, “Quasicyclic low-density parity-check codes from circulant permutation matrices,’ *IEEE Trans. Inf. Theory*, vol. 50, no. 8, pp. 1788-1793, Aug. 2004.

[24] K. Mazooji, F. Sala, G. Van den Broeck, and L. Dolecek, “Robust channel coding strategies for machine learning data,” in *Proc. of 54nd Annual Allerton Conference on Communication, Control, and Computing*, Allerton, IL, Sep. 2016.

[25] F. Sala, S. Kabir, G. Van den Broeck, and L. Dolecek, “Don’t Fear the Bit Flips: Optimized Coding Strategies for Binary Classification,” arXiv preprint arXiv:1703.02641 (2017).

[26] F. Sala, S. Kabir, G. Van den Broeck, and L. Dolecek, “Don’t Fear the Bit Flips: Robust Linear Prediction Through Informed Channel Coding,” in *Reliable machine learning in the wild (WILDML 2017)*, August 2017.

[27] E. Kalapanidas, N. Avouris, M. Craciun, and D. Neagu, “Machine learning algorithms: a study on noise sensitivity,” in *Proc. 1st Balkan Conference in Informatics*, Thessaloniki, Greece, Nov. 2003.

[28] O. Dekel, and O. Shamir, “Learning to classify with missing and corrupted features, ” in *Proc. 25th Int. Conf. on Machine Learning*, Helsinki, Finland Jul. 2008.

[29] A. Globerson, and S. Roweis, “Nightmare at test time: robust learning by feature deletion,” in *Proc. 23rd Int. Conf. on Machine Learning*, Pittsburgh, PA, Jun. 2006.

[30] A. Choi, Y. Xue, and A. Darwiche, “Same-decision probability: A confidence measure for threshold-based decisions,” *Int. J. of Approximate Reasoning* vol. 53, no. 9, pp. 1415-1428, Dec. 2012.

[31] B. Frenay, and A. Kaban, “A comprehensive introduction to label noise,” in *Proc. European Symp. on Artificial Neural Networks, Comp. Intelligence and Machine Learning, ESANN*, Bruges, Belgium, Apr. 2014.

[32] C. M. Riggle, and S. G. McCarthy, “Design of error correction systems for disk drives,” *IEEE Trans. Magn*, vol 34, no. 4, pp. 2362-2371, Jul. 1998.

[33] Y. Cassuto, M. Schwartz, V. Bohossian, and J. Bruck, “Codes for asymmetric limited-magnitude errors with application to multilevel flash memories,” *IEEE Trans. Inf. Theory*, vol 56, no. 4, pp. 1582-95, Apr. 2010.

[34] L. Dolecek, and F. Sala, “Channel coding methods for non-volatile memories,” *Foundations and Trends in Communications and Information Theory*, vol. 13, no. 1, pp. 1-28, Feb 2016.

[35] K. Zhao, W. Zhao, H. Sun, T. Zhang, X. Zhang, and N. Zheng, “LDPC-in-SSD: making advanced error correction codes work effectively in solid state drives,” in *Proc of Conf. on File and Storage Technologies (FAST ’13)*, San Jose, CA Feb. 2013.

[36] G. Balakrishnan, M. Yang, Y. Jiang, Y. Kim, "Performance analysis of error control codes for wireless sensor networks," in *Proc. IEEE Int. Conf. on Inf Theory*, Las Vegas, NV, Apr. 2007.

[37] S. Chung, and S. Lee, "Coding for Wireless Sensor Networks," in *Smart Sensors for Health and Environment Monitoring*, Springer. Netherlands, 2015, pp 307-323.

[38] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient Coding: Avoiding Stragglers in Synchronous Gradient Descent," *stat.*, vol. 1050, Mar. 2017.

[39] N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran, "Distributed storage codes with repair-by-transfer and nonachievability of interior points on the storage-bandwidth tradeoff," in *IEEE Trans. Inf. Theory*, vol 58, no. 3, pp. 1837-52, Mar. 2012.

[40] A. Krause, and D. Golovin.(2014) *Submodular function maximization*[Online]. Available: `http://www.cs.cmu.edu/afs/.cs.cmu.edu/Web/People/dgolovin/papers/submodular_survey12.pdf`.

[41] J. Wishart, "The generalised product moment distribution in samples from a normal multivariate population," *Biometrika*, pp. 32-52, Jul. 1928.

[42] A. Laub, *Matrix analysis for scientists and engineers.* SIAM, 2005.

[43] L. N. Trefethen, *Numerical linear algebra.* SIAM, 1997.

[44] P. Tufekci, "Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods," *Int. J. of Elect. Power and Energy Syst.*, vol. 60, pp. 126-140, Sep. 2014.

[45] F. Sala, C. Schoeny, S. Kabir, D. Divsalar, and L. Dolecek, "On nonuniform noisy decoding for LDPC codes with application to radiation-induced errors," *IEEE Trans. Commun*, vol. 65, no. 4, pp. 1438-1450, Apr. 2017.