Aspects of Convex Optimization and Concentration in Coding

Ronen Eshel

Aspects of Convex Optimization and Concentration in Coding

Research Thesis

Submitted in Partial Fulfillment of the Requirements
For the Degree of Master of Science
in Electrical Engineering

Ronen Eshel

Submitted to the Senate of the Technion - Israel Institute of Technology
SHEBAT 5772 HAIFA FEBRUARY 2012

The Research Thesis Was Done Under the Supervision of Prof. Igal Sason in the Electrical Engineering Department

ACKNOWLEDGMENTS

My sincere and wholehearted thanks go, first and foremost, to my supervisor, Prof. Igal Sason. Without his meticulous and consistent devotion to my guidance, this work would not be. He has also taught me a great deal about the fine art of academic writing. Finally, I thank my parents, for their moral support and never-ending patience throughout this period. This thesis is devoted to them.

The Generous Financial Help Of The Technion Is Gratefully Acknowledged.

To my parents, Ruth and Reuven.

Contents

List of Figures

Abstract

The thesis consists of three main parts. The first part of the work considers the complexity of the interior-point method (IPM) for solving convex optimization problems. We prove upper bounds on the number of Newton iterations needed to solve a convex optimization problem whose objective function is self concordant (s.c.). Several bounds are given for various step size algorithms: Newton's method with backtracking line-search and two choices of pre-determined step size. Compared to previous bounds, the new bounds are 10-100 times tighter. Using computer simulations, we explore the properties of those new tightened bounds.

The second part of the thesis considers the problem of decoding linear block codes (e.g., low-density parity-check codes (LDPC)) using Linear Programming (LP). We apply the bounds derived earlier on an IPM-based LP decoder in order to obtain a complexity bound on the number of iterations of the LP decoder. Next, we optimize the bound in order to obtain an optimized set of IPM parameters based on the code and channel parameters (i.e., block length, parity-check matrix row degree, noise level, etc..). The bound derived gives an analytic insight on the decoding complexity as a function of the code and channel parameters.

The third part of the work (which stands by itself) considers the concentration of measures for LDPC code ensembles. The results derived in this thesis follow from Azuma-Hoeffding inequality for Doob's martingales with bounded differences. The first result is a tightened concentration inequality for the conditional entropy (originally derived by Méasson et al.). Next, several concentration results on the number of erroneous variable-to-check node messages for inter-symbol-interference (ISI) channels are proved. The analysis provides an explicit expression for the exponential decay of the concentration inequalities given by Kavcić et al. It is shown that specializing these results for memoryless channels provides tightened concentration inequalities

as compared to previous results by Luby et al. and Richardson & Urbanke.

List of notation and abbreviations

s.c.	:self-concordant
IPM	:interior-point method
UCSCM	:Un-constrained s.c. minimization
ECSCM	:Equality-constrained s.c. minimization
IECSCM	:Inequality and equality constrained s.c. minimization
$\lambda^{(n)}$	:The Newton decrement at the n-th Newton iteration
KKT	:Karush–Kuhn–Tucker (conditions or matrix)
Δx_{nt}	:The Newton step
α	:backtracking line search parameter
β	:backtracking line search parameter
μ	:IPM search parameter
ϵ	:minimization problem tolerance
p^*	:minimization problem's solution
ISI	:Inter-Symbol-Interference
MBIOS	:Memoryless, Binary-Input and Output-Symmetric
AWGN	:Additive White Gaussian Noise
BIAWGN	:Binary Input Additive White Gaussian Noise
C	:Capacity of the channel
SNR	:Signal to Noise Ratio
BEC	:Binary Erasure Channel
BSC	:Binary Symmetric Channel
BER	:Bit Error Rate
WER	:Word Error Rate
i.i.d.	:independent identically distributed
i.u.d.	:independent uniformly distributed

LLR	:Log Likelihood Ratio
$\ell(y)$	:LLR of received word y
g_p	:The pdf of the LLR at the output of an MBIOS channel
LDPC	:Low-Density Parity-Check
$\ell_{\max}$	:Maximal absolute value of the log-likelihood-ratio vector
ML	:Maximum Likelihood
MLD	:Maximum Likelihood Decoder
$\mathbf{y}_{\mathrm{ML}}$	:The Maximum likelihood word
LP	:Linear Programming
d_{c}	:LDPC code's check node degree
d_{v}	:LDPC variable node degree
$\mathcal{P}(H)$	:Fundamental polytope associated with parity check matrix H
Ω	:Probability space
$\mathbb{P}(A)$	:Probability of event A
RV	:Random Vector
$h_2(\cdot)$	:Binary entropy function
$H(\mathbf{X}\|\mathbf{Y})$	:The conditional entropy of vector $\mathbf{X}$ given $\mathbf{Y}$
k	:The dimension of a linear block code
n	:The length of a block code
R	:Code rate
$\mathcal{C}$	:Code-book
H	:Parity check matrix of a linear code
G	:Generator matrix of a linear code
$\mathcal{G}$	:The factor graph of a code
$N^{(\ell)}{}_{\vec{e}}$	:Neighborhood of depth ℓ of an edge $\vec{e}$
$N_{\mathrm{c}}^{(\ell)}$	:Total number of check-nodes in the neighborhood $\mathcal{N}^{(\ell)}{}_{\vec{e}}$
$N_{\mathrm{v}}^{(\ell)}$	:Total number of variable-nodes in the neighborhood $\mathcal{N}^{(\ell)}{}_{\vec{e}}$
$N_{\mathrm{e}}^{(\ell)}$	:Total number of code related edges in the neighborhood $\mathcal{N}^{(\ell)}{}_{\vec{e}}$

Chapter 1

Introduction

Low-density parity-check (LDPC) codes were discovered by Gallager in 1962 [42]. In the 1990s, they were rediscovered by a number of researchers ([12],[34],[38]) and since then, the subject has become a central research theme in coding theory. Research in this field was followed by the adoption of those codes in commercial communication systems. The performance of these codes is extraordinary and their complexity is moderate, thus making them feasible for practical use. In 2001, Chung et al. [46] have demonstrated a family of LDPC codes whose bit error rate vanishes at $\frac{E_b}{N_0}$ that is 0.0045 dB from the capacity of the binary-input AWGN channel, as the block length tends to infinity. One of the challenges that face researchers in this field is analyzing the behavior of these codes. Current analysis is usually based on a method called *density-evolution* who predicts the asymptotic performance of LDPC code ensembles. Though useful, this method is still limited, as analytic results are known only for simple channels (e.g. BEC and BSC). Other channels require numerical calculations which provide only numerical results with the ability to derive closed-form expressions for capacity-approaching ensembles. The analysis is especially difficult when the factor graph contains cycles or if short to moderate block lengths are considered.

1.1 Convex optimization and LP decoding of linear codes

The decoders most often used for LDPC codes are based on the *belief-propagation* algorithm [43](e.g., sum-product algorithm), where messages are iteratively sent across a factor graph of the code. In recent years, the idea of linear programming (LP) has attracted the attention of many researchers. This alternative decoding algorithm is based on the fact that Maximum-Likelihood Decoding (MLD) can be seen as a combinatorial optimization problem. In order to make the complexity feasible, a relaxation technique is used. The idea is to relax the definition of the feasible set from a discrete set to a continuous subset in $\mathbb{R}^n$. The elimination of the constraint that a feasible point is integral leads to a linear programming (LP) problem that can be efficiently solved. The resulting optimization problem might have a different optima (e.g., a non-integral solution called a *pseudo-codeword*) with respect to the original problem, however its computational complexity is drastically reduced, thus providing a trade-off between the decoding complexity and performance. The research conducted by Feldman et al. ([21], [22]) was the first to apply the concept of relaxation to decoding problems. Since then, numerous studies on LP decoding have been conducted ([2], [10], [25], [30], [31], [40], [41]). An LP decoder for LDPC codes is an optimization problem characterized with many constraints and variables. Therefore, in order to make LP decoding feasible for codes with long block length, efficient LP solvers are required. During the last twenty years, there has been a revolution in the methods used to solve optimization problems. In the early 1980s, sequential quadratic programming and augmented Lagrangian methods were favored for nonlinear problems, while the simplex method was basically unchallenged for linear programming. Since then, modern interior-point methods (IPMs) ([5], [9], [45]) have infused virtually every area of continuous optimization, and have forced great improvements in the earlier methods. In 1984, N. Karmarkar [37] discovered a polynomial interior-point method that is practically more efficient than the ellipsoid method. He also claimed superior performance compared to the simplex method. At the same time, Nesterov and Nemirovski investigated the new methods from a more fundamental viewpoint: What are the basic properties that lead to polynomial-time complexity? It turned out that the key property is that the barrier function should be self-concordant. This seemed

to provide a clear, complexity-based criterion to delineate the class of optimization problems that could be solved in a provably efficient way using the new methods. The culmination of this work was the book by Nesterov and Nemirovski (1994) [52], whose complexity emphasis contrasted with the classic text on barrier methods by Fiacco and McCormick (1968). These advances in optimization theory made problems like LP decoders computationally tractable. Using the self-concordant property, much can be said about the complexity of the LP decoder (e.g., it can be shown that the computational time grow moderately with accuracy and dimension of the problem). In this thesis, we prove complexity bounds on self-concordant functions, and apply those bounds to derive bounds on the complexity of the LP decoder.

1.2 Concentration of measures in LDPC code ensembles

The concentration of measures phenomenon was first put forward during the seventies and eighties in geometric functional analysis, and has been subject to very nice developments in probability theory, mostly due to Ledoux [32] and Talagrand [35]. Very roughly speaking, this phenomenon can be stated in the following simple way: "A random variable that depends in a smooth way on many independent random variables (but not too much on any of them) is essentially constant". The exact meaning of such a statement clearly needs to be clarified rigorously, but it will often mean that such a random variable X concentrates around a constant c (where c denotes either the statistical expectation or median of X) in a way that the probability of the event $\{|X - c| > t\}$ decays exponentially in t (for $t \geq 0$). This type of bound is referred to as a *concentration inequality*. The mathematical foundations of concentration of measures are considered, e.g., in [7], [32] and [36, Chapter 7]. Concentration of measures inequalities are also at the core of probabilistic analysis of randomized algorithms (see, e.g., [4], [13], [44]). The concentration of measure phenomenon is a principle that is applied in measure theory, probability and combinatorics, and it has consequences in some other fields such as functional analysis (see, e.g., [13], [32] and [35]). The basic concentration theorem of iterative message-passing decoding (see [49,

pp. 487–490]) asserts that all except an exponentially (in the block length) small fraction of codes perform within an arbitrary small δ from the ensemble average (where δ is a positive number that can be chosen arbitrarily small). Therefore, assuming a sufficiently long block length, the ensemble average forms a good indicator for the performance of individual codes. In general, all the concentration inequalities which have been proved in the setting of iterative message-passing decoding so far are rather loose, and much stronger concentration phenomena can be observed in practice for moderate to long block lengths. Therefore, to date, these concentration inequalities serve mostly to justify theoretically the ensemble approach, but they are not tight for finite block lengths. In this thesis, we provide concentration of measures of LDPC code ensemble and tighten some known results.

1.3 Thesis outline

The thesis is composed of three main chapters and a concluding chapter.

1. **Chapter 2 [Complexity analysis of convex optimization problems solved using interior-point method]**

   - We provide the reader with some background material about convex optimization. Classic algorithms such as Newton's method and interior-point method (IPM) are explained. Next, we define the property of self-concordance which is used to analyze the convergence rate of Newton's method and the IPM.
   - A convergence rate analysis of Newton's method is conducted. New tightened upper bounds on the number of Newton iterations are proved. These bounds are given for the case of backtracking line-search and pre-determined step size. Next, we extend the use of these bounds to the IPM.
   - Numerical simulations are performed in order to explore the behavior of the bounds derived.

2. **Chapter 3 [Complexity analysis of IPM-based LP decoders]**

   - We provide the reader with some background material about LP decoding of linear codes.

- An LP decoder which is based on the IPM is introduced. The convergence rate analysis derived in the previous chapter is applied to the LP decoder. These bounds are used to study the complexity of the IPM-based LP decoder
- We compare the new tightened bounds derived to some previously reported bounds.

3. **Chapter 4 [Concentration of measures in LDPC code Ensembles]**

   - We present relevant mathematical background that is essential for the analysis in this work, and also provide briefly some applications of these mathematical tools in coding and communication theory.
   - A large-deviation analysis of the conditional entropy is provided, and tightened concentration inequalities are derived. The latter inequalities are compared to the original bound by Méasson et al. [8, Theorem 4].
   - Concentration results on the number of erroneous variable-to-check node messages are derived for Inter-Symbol-Interference (ISI) channels. The analysis provides explicit expression for the exponential rate that is related to the concentration inequalities in [3, Theorems 1 and 2]. It is shown that particularizing these results for memoryless channels provides tightened concentration inequalities as compared to [29] and [48].

4. **Chapter 5 [Summary, conclusions and future research directions]** We conclude our work in Section 5.1, where we summarize the main results and conclusions of the thesis. Finally, in Section 5.2, a number of directions are suggested for future research in this area.

Chapter 2

Complexity analysis of convex optimization problems solved using interior-point method

2.1 Short overview

In the following, we consider the interior-point method (IPM) for solving convex optimization problems. The main purpose of this chapter is to provide a tightened complexity analysis for this method. Since the IPM is an iterative algorithm, the complexity of the solver is presented as a bound on the number of Newton iterations. In Chapter 3, we shall use this analysis to provide bounds on the complexity of an IPM-based LP decoder which is a relaxed version of the ML decoder.

2.2 Convex optimization and interior-point method

In this section, we provide some basic terminology and notation that is related to convex optimization and the interior-point method.

Definition 1 [Convex optimization problem]
A convex optimization problem is an optimization problem of the form

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, ..., m \\ & Ax = b \end{array} \tag{2.1}$$

where $f_0, ..., f_m : \mathbb{R}^n \to \mathbb{R}$ are convex functions, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^{m \times 1}$,.

We note that the feasible set of a convex optimization problem is convex. We call the problem feasible if an optimal solution x^* exists, and denote p^* as the optimal value, $p^* = \inf\{f(x) | \text{constraints}\} = f(x^*)$.

A particular example of a convex optimization problem is the family of Linear Programming (LP) problems. The problem is called a linear program if the objective and constraint functions are all affine.

Definition 2 [Linear programming optimization problem]

$$\begin{array}{ll} \text{minimize} & c^T x + d \\ \text{subject to} & Gx \preccurlyeq h \\ & Ax = b \end{array} \tag{2.2}$$

Where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^{m \times 1}$, $G \in \mathbb{R}^{p \times n}$, $h \in \mathbb{R}^{p \times 1}$, $c \in \mathbb{R}^{n \times 1}$ and $d \in \mathbb{R}$.

The geometric interpretation of an LP is illustrated in Figure 2.1. The feasible set of the LP problem is a polyhedron $\mathcal{P}$. The objective is to minimize the affine function $c^T x + d$ over $\mathcal{P}$. The objective function $c^T x + d$ is affine, so its level curves are hyper-planes orthogonal to c (shown as dashed lines). The point x^* is the optimal point, and geometrically it is the point in $\mathcal{P}$ as far as possible in the direction $-c$.

Remark 1 It can be shown that if a single solution exists, then it is a vertex of the polyhedron $x^* \in \mathcal{V}(\mathcal{P})$. Therefore, if the polyhedron $\mathcal{P}$ is the convex hull of a set $\mathcal{C}$ (i.e.,$\mathcal{P} = \text{Conv}(\mathcal{C})$), then if a single solution x^* exist, then $x^* \in \mathcal{C}$

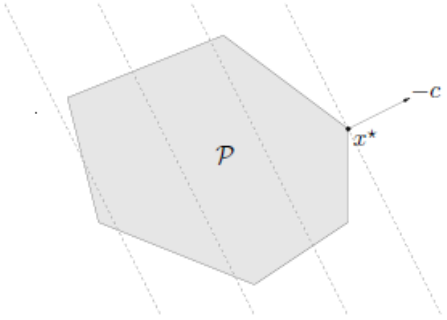

Figure 2.1: The geometric interpretation of an LP. The feasible set $\mathcal{P}$, which is a polyhedron, is shaded. The objective function $c^T x + d$ is affine, so its level curves are hyper-planes orthogonal to c (shown as dashed lines). The point x^* is optimal and geometrically it is the point in $\mathcal{P}$ as far as possible in the direction $-c$.

2.2.1 Minimization of un-constrained and equality-constrained convex problems

Consider Definition 1 of a convex optimization problem. If $m = 0$ and $p > 0$ we say the problem is *equality-constrained*, while if $m = p = 0$ we say the problem is *un-constrained*. Next, we present Newton's method, which is an efficient method for solving general convex optimization problems.

Newton's method for un-constrained minimization

Newton's method is an iterative algorithm for solving convex optimization problems. The idea behind the algorithm is to compute the minimizer of the second order polynomial approximation of the objective function. The resulting minimizer is used as a starting point for another iteration of the Newton's method.

Definition 3 [The Newton's method for un-constrained minimization problems]
Consider an un-constrained convex optimization problem. Given a starting point $x^{(0)} \in \mathbf{dom}(f)$, tolerance $\epsilon > 0$, repeat

1. *Compute the Newton step* : $\Delta x_{nt} = -\nabla^2 f(x)^{-1} \nabla f(x)$.

2. *Stopping criterion* : Compute the Newton decrement $\lambda^2 = \Delta x_{nt}^T \nabla^2 f(x) \Delta x_{nt}$. Quit if $\lambda^2/2 \leq \epsilon$.

3. *Line search* : Choose step size t. (e.g., using backtracking line-search)

4. *Update* : $x := x + t\Delta x_{nt}$

Newton's method, as outlined above, is sometimes called the damped Newton method or guarded Newton method, to distinguish it from the pure Newton method, which uses a fixed step size t = 1. In the algorithm described above, the search direction is the vector Δx_{nt}, however the step size is scaled by factor t which is calculated using a line-search algorithm. We shall consider the following line-search algorithms

1. Exact line-search : The optimal value $t_{\text{Exact}} = \arg\min f(x + t\Delta x_{nt})$ is used. This algorithm requires the lowest number of Newton iterations but it is computationally complex because of the minimization required to calculate t.

2. Backtracking line-search : A low complexity line-search algorithm that approximates the exact line-search. This algorithm is detailed later in this subsection.

3. Predetermined or fixed step size : The step size is a predetermined function of the current state of variables (e.g.,$\lambda(x), x, |\nabla f(x)|$...). In general, it offers the lowest complexity with the tradeoff of slower convergence rate.

The following are the main properties of the Newton's method

1. Invariant to linear scaling : If we apply a non-singular linear transformation $y = Tx$, then the iterates when minimizing $\tilde{f}(y) = f(Tx)$ are related by the same change of coordinates (i.e., $Ty^{(k)} = x^{(k)}$).

2. The tolerance can be approximated using the Newton decrement

$$f(x) - p^* \approx \lambda^2/2.$$

   This approximation is highly accurate for $\lambda \ll 1$.

3. The number of Newton iterations increases slowly as the problem size n increases.

Newton's method for Equality-constrained minimization

Newton's method described above, can be extended to include equality constraints. The method is almost the same as the Newton's method without constraints except for two differences

1. The initial point must be feasible (i.e., satisfy $x^{(0)} \in \mathbf{dom}(f)$ and $Ax = b$)

2. The definition of the Newton step is modified to take the equality constraints into account. In particular, we make sure that the Newton step Δx_{nt} is a feasible direction (i.e., $A\Delta x_{nt} = 0$). The Newton step Δx_{nt} is computed by solving the so called KKT equations

$$\begin{bmatrix} \nabla^2 f(x) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x_{nt} \\ \omega \end{bmatrix} = \begin{bmatrix} -\nabla f(x) \\ 0 \end{bmatrix}.$$

Besides those modifications, the algorithm is the same and has similar properties.

Backtracking line-search

As mentioned before, the exact line-search computes the optimal step size which in turn yields the lowest possible number of Newton iterations. However computing the exact line-search involves solving another optimization problem, thus it is usually computationally complex. Therefore, most line-searches used in practice are non-exact. The step length is chosen to approximately minimize $f(x)$ along the ray $\{x + t\Delta x | t \geq 0\}$, or even to just reduce $f(x)$ enough.

One non-exact line-search method that is very simple and quite effective is called backtracking line-search. It depends on two backtracking line-search parameters $\alpha \in (0, 0.5)$ and $\beta \in (0, 1)$.

Definition 4 [The Backtracking line-search]
Given the line-search parameters $\alpha \in (0, 0.5), \beta \in (0, 1)$, and a descent direction Δx for $f(x)$ at $x \in \mathbf{dom}(f)$, initialize t with $t := 1$ and preform the following iterative algorithm :

1. If $f(x + t\Delta x) \leq f(x) + \alpha t \nabla f(x)^T \Delta x$, quit.

2. Update $t := \beta t$

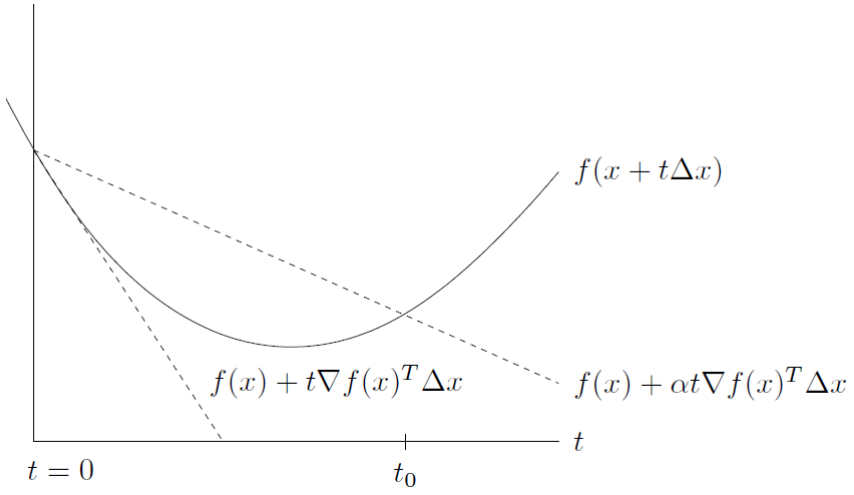

Figure 2.2: *Backtracking line-search.* The curve shows f, restricted to the line over which we search. The lower dashed line shows the linear extrapolation of f, and the upper dashed line has a slope a factor of α smaller. The backtracking condition is that f lies below the upper dashed line, i.e., $0 \leq t \leq t_0$.

The line-search is called backtracking because it starts with unit step size and then reduces it by factor β until the stopping condition holds. Since Δx is a descent direction, we have $\nabla f(x)^T \Delta x < 0$, so for small enough t we have $f(x + t\Delta x) \approx f(x) + t\nabla f(x)^T \Delta x < f(x) + \alpha t \nabla f(x)^T \Delta x$, which shows that the backtracking line-search eventually terminates. As illustrated in Figure 2.2, the constant α can be interpreted as the fraction of the decrease in f predicted by linear extrapolation that we will accept.

2.2.2 Inequality-constrained convex problems - interior-point methods

In this subsection, we discuss interior-point methods for solving convex optimization problems that include inequality constraints and affine equality constraints

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i \leq 0, \quad i = 1, ..., m \\ & Ax = b \end{array} \tag{2.3}$$

Where $f_0, ..., f_m : \mathbb{R}^n \to \mathbb{R}$ are convex and twice continuously differentiable, and $A \in \mathbb{R}^{p \times n}$ with $\text{rank}(A) = p < n$. We assume that an optimal solution x^* exists and denote $p^* = \inf\{f(x) | \, Ax = b\} = f(x^*)$ as the optimal value.

The key concept of interior-point methods is to solve (2.3) by applying Newton's method to a sequence of equality-constrained problems. We will focus on a particular interior-point algorithm called the logarithmic barrier method.

Our goal is to approximately formulate the inequality-constrained problem (2.3) as an equality-constrained problem to which Newton's method can be applied. The first step is to rewrite problem (2.3), making the inequality constraints implicit in the objective function:

$$\begin{array}{ll} \text{minimize} & f_0(x) + \sum_{i=1}^{m} I\left(f_i(x)\right) \\ \text{subject to} & Ax = b \end{array} \tag{2.4}$$

where $I(x)$ is an indicator function, i.e.,

$$I(x) = \begin{cases} 0 & x \leq 0 \\ \infty & x > 0. \end{cases}$$

The indicator function can be approximated by a logarithmic barrier function with parameter $t > 0$

$$I^{(t)}(x) = -(1/t)\log(-x).$$

As t increases, the approximation becomes more accurate. We denote $x^*(t)$ as the minimizer of the equality-constrained problem with the logarithmic barrier function $I^{(t)}(x)$ replacing the indicator function. It can be shown that this point is at most m/t-suboptimal, i.e.,

$$f_0(x^*(t)) - p^* \leq m/t.$$

The barrier method is based on solving a sequence of equality-constrained minimization problems with increasing value of the parameter t. At each iteration the last point found is used as a starting point for the next minimization problem. In other words, we compute $x^*(t)$ for a sequence of increasing values of t (this sequence is called the *central path*), until $t \geq m/\epsilon$, which guarantees that we have an ϵ-suboptimal solution of the original problem.

Definition 5 [The barrier method]
Consider the convex optimization problem given in (2.3) and define the barrier function $B(x) = -\sum_{i=1}^{m} \log(-f_i(x))$. Given a strictly feasible starting point $x^{(0)}$, search parameters $t := t^{(0))}$, $\mu > 1$ and required tolerance $\epsilon > 0$. Repeat the following steps

1. Centering step : Compute $x^*(t)$ by minimizing $tf_0(x)+B(x)$, subject to $Ax = b$, starting at x.
2. Update starting point : $x := x^*(t)$.
3. Stopping criterion : Quit if $m/t < \epsilon$.
4. Increase t : $t := \mu t$.

As seen in Figure 2.3, the algorithm involves two layers of iterations. The iterations along the central-path are called *outer iterations*. Within each outer iteration, Newton's method is applied to find the minimum of $tf_0(x)+B(x)$, subject to $Ax = b$. The Newton iterations are called *inner iterations*.

Remark 2 The choice of the parameter μ involves a trade-off between the number of inner and outer iterations. If μ is small, then many outer iterations are required. However, each inner iteration will be shorter because the result from the last outer iteration is a good starting point for the next one. If μ is large the opposite is true. In practice for μ in the range 3 to 100 the two effects nearly cancel, making the choice of μ not critical.

2.2.3 Complexity analysis using self-concordance property

When considering general convex problems, not much can be said about the complexity of the solved problem. The classical convergence analysis of Newton's method

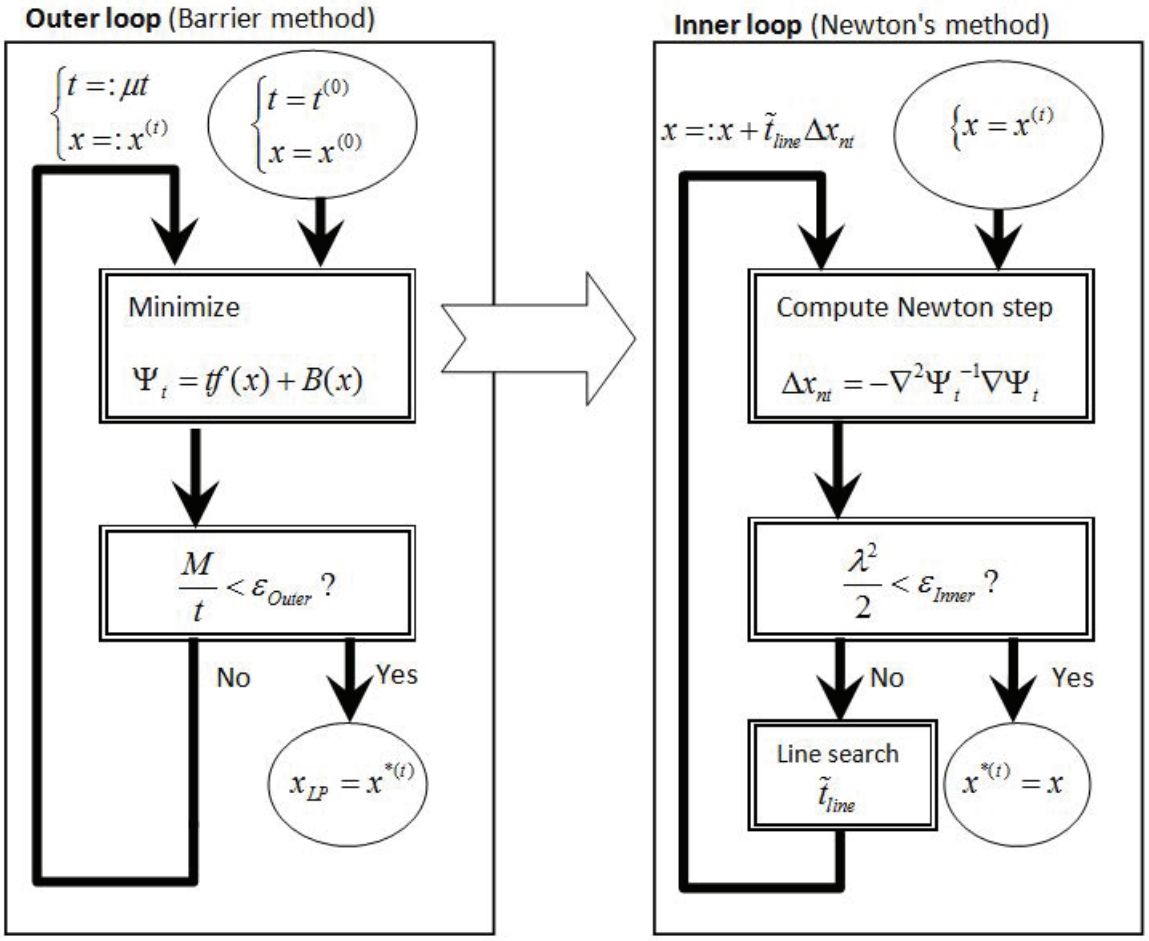

Figure 2.3: The figure illustrates the two layers of the interior-point method. The iterations along the central-path are called *outer iterations*. Within each outer iteration, Newton's method is applied to find the minimum of $tf_0(x) + B(x)$, subject to $Ax = b$. The Newton iterations are called *inner iterations*.

depends on constants that characterize the minimized function. However, in practice those constants are usually not known or hard to calculate. Nesterov and Nemirovski [52] discovered that if we restrict ourselves to a family of convex functions called 'self-concordant' functions, a simple and elegant complexity analysis can be obtained. This family is important for several reasons

- The logarithmic barrier functions are included in this family.
- The complexity analysis of this family includes only known parameters.
- Like Newton's method, the property of self-concordance is affine-invariant.

Definition 6 [Self-concordant function (s.c.)]
A convex function $f : \mathbb{R} \to \mathbb{R}$ is self-concordant (s.c.) if

$$|f^{(3)}(x)| \leq 2f''(x)^{3/2} \tag{2.5}$$

for all $x \in \mathbf{dom}(f)$. A function $f : \mathbb{R}^n \to \mathbb{R}$, with $n > 1$, is s.c. if it is s.c. along every line in its domain (i.e., if the function $\tilde{f}(t) = f(x + tv)$ is a s.c function of t for every point $x \in \mathbf{dom}(f)$ and $v \in \mathbb{R}$ such that $x + tv \in \mathbf{dom}(f)$).

Self-concordance calculus

The property of s.c. is maintained over some basic mathematical operations.

1. Scaling - If f is s.c. and $a \geq 1$ then af is also s.c..
2. Sum - If $f_1(x), f_2(x)$ are s.c. then $f_1(x) + f_2(x)$ is also s.c..
3. Composition with affine function - If f is s.c. and $T(x) = Ax + b$ is an affine transformation then $f(Ax + b)$ is also s.c..
4. Composition with logarithm function - Let $g : \mathbb{R} \to \mathbb{R}$ be a convex function with $\mathbf{dom}(g) = \mathbb{R}_{++}$ and
$$|g'''(x)| \leq \frac{3g''(x)}{x}.$$
Then $f(x) = -\log(-g(x)) - \log(x)$ is s.c. on $\{x | x > 0, g(x) < 0\}$.

Self-concordance related minimization problems

We define notations for some basic s.c. related optimization problems.

1. **Definition 7 [Un-constrained s.c. minimization (UCSCM) problem)]** An un-constrained convex optimization problem

$$\text{minimize } f(x) \tag{2.6}$$

Where $f : \mathbb{R}^n \to \mathbb{R}$ is an s.c. function. We assume that an optimal solution x^* exists, and let p^* denote the optimal value, $p^* = \inf f(x) = f(x^*)$.

2. **Definition 8 [Equality-constrained s.c. minimization (ECSCM) problem)]**
A convex optimization problem with affine equality constraints

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \end{array} \tag{2.7}$$

Where $f : \mathbb{R}^n \to \mathbb{R}$ is an s.c. function, and $A \in \mathbb{R}^{m \times n}$ with rank$A = m < n$. We assume that an optimal solution x^* exists, and let p^* denote the optimal value, $p^* = \inf\{f(x) | \, Ax = b\} = f(x^*)$.

3. **Definition 9 [Inequality and equality constrained s.c. minimization (IECSCM) problem)]**
A convex optimization problem with affine equality constraints and convex inequality constraints.

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & Ax = b \\ & f_i\,(x) \leq 0 \quad i = 1, ..., m \end{array} \tag{2.8}$$

Where $f_0 : \mathbb{R}^n \to \mathbb{R}$ is a s.c. function, $A \in \mathbb{R}^{p \times n}$ has rank $p < n$ and $f_0, ..., f_m : R^n \to R$ are convex and twice differentiable.

Complexity and optimality bounds using the property of self-concordance

In this subsection, complexity and optimality bounds related to the s.c. property are provided.

Theorem 1 [**Bound on sub-optimality**] Consider an ECSCM problem (or the degenerated UCSCM problem), denote p^* as the optimal value and $\lambda(x)$ as the Newton's decrement. If $\lambda(x) \leq 0.68$, then the following inequality holds

$$\lambda(x)^2 \geq f(x) - p^*$$

Proof: See Sections 9.6.3 and 10.2.3 in [45].

Remark 3 If $\epsilon < 0.68^2$ is the required tolerance, we can use $\lambda(x)^2 < \epsilon$ as the stopping criterion to guarantee that on exit $f(x) - p^* \leq \epsilon$.

Theorem 2 [**Complexity bound for ECSCM problems**] Consider an ECSCM problem (or the degenerated UCSCM problem) solved using Newton's method with backtracking line-search. Denote p^* as the optimal value, ϵ as the required tolerance, β and α as the backtracking line-search parameters and $x^{(0)}$ as the starting point. Then, the number of Newton iterations is bounded by

$$N_{\text{Total}} \leq \left(\frac{20 - 8\alpha}{\alpha\beta(1-2\alpha)^2} \right) \left(f(x^{(0)}) - p^* \right) + \log_2 \log_2 \left(\frac{1}{\varepsilon} \right).$$

Proof: See Section 2.3 for an outline of the proof. The full proof is given in Sections 9.6.4 and 10.2.4 in [45].

Theorem 3 [**Complexity bound for IECSCM problems**] Consider an IECSCM optimization problem. Let the problem be solved using the IPM with Newton's method and backtracking line-search. Set the parameters of the outer iterations to $t^{(0)}, \mu$, and the parameters of the backtracking line-search to α, β. Then, the total number of Newton iteration (excluding the initial centering step) is upper bounded by

$$N_{\text{Total}}^{\text{Inequality}} \leq \left\lceil \frac{\log\left(m/(\varepsilon t^{(0)})\right)}{\log \mu} \right\rceil \left(\left(\frac{20 - 8\alpha}{\alpha\beta(1-2\alpha)^2} \right) \left(m\left(\mu - 1 - \log \mu\right) \right) + \log_2 \log_2 \left(\frac{1}{\varepsilon} \right) \right)$$

Proof: See Section 11.5.2 in [45].

2.3 Revision of the classical analysis of the convergence rate of Newton's method

The proofs of the Theorems given in Section 2.4 are highly related to the proof of Theorem 2. Therefore, we revisit the classical analysis of Newton's method with backtracking line-search when applied to an ECSCM problem. The analysis is given for the un-constrained UCSCM problem and then shown to also apply for the constrained problem. We assume that f is bounded below and a starting point $x^{(0)}$ is set for the minimization of f with Newton's method. The sub-level set $S = \{x|f(x) \leq f(x^{(0)})\}$ is also assumed to be closed. In the following, we show that there exist some positive numbers η and γ, where $0 < \eta \leq 1/4$ depends only on the line-search parameters α and β, such that the following holds:

- [**Damped phase**] - If $\lambda(x^{(k)}) > \eta$, then for some $\gamma > 0$ (independent of k)

$$f(x^{(k+1)}) - f(x^{(k)}) \leq -\gamma \tag{2.9}$$

- [**Quadratic phase**] - If $\lambda(x^{(k)}) \leq \eta$ then the backtracking line-search selects $t = 1$ and

$$2\lambda(x^{(k+1)}) \leq \big(2\lambda(x^{(k)})\big)^2. \tag{2.10}$$

Inequality (2.10) can be applied recursively, to conclude that

$$\lambda(x^{(l)}) \leq \eta, \quad \forall\, l \geq k$$

and

$$2\lambda(x^{(k+1)}) \leq \left(2\lambda(x^{(k)})\right)^2 \leq (2\eta)^{2^{l-k}} \leq \left(\frac{1}{2}\right)^{2^{l-k}}.$$

Using Theorem 1, the following inequality holds

$$f(x^{(l)}) - p^* \leq \lambda(x^{(l)})^2, \quad \lambda \leq 0.68.$$

As a consequence, for all $l \geq k$

$$f\big(x^{(l)}\big) - p^* \leq \lambda\big(x^{(l)}\big)^2 \leq \frac{1}{4}\left(\frac{1}{2}\right)^{2^{l-k+1}} < \left(\frac{1}{2}\right)^{2^{l-k+1}}$$

and hence $f(x^{(k)}) - p^* \leq \varepsilon$ if $l - k \geq \log_2 \log_2(1/\varepsilon)$.

Inequality (2.9) implies that the damped Newton phase requires not more than $(f(x^{(0)}) - p^*)/\gamma$ iterations. Thus, the total number of iterations that is required for the calculation of the minimum of f with accuracy ε, starting at a point $x^{(0)}$, is upper bounded by

$$\frac{f(x^{(0)}) - p^*}{\gamma} + \log_2 \log_2 \frac{1}{\varepsilon}. \tag{2.11}$$

2.3.1 Damped Newton Phase

If we let $\tilde{f}(t) = f(x + t\Delta x_{nt})$, then we have

$$\tilde{f}'(0) = -\lambda(x)^2, \quad \tilde{f}''(0) = \lambda(x)^2.$$

The property of s.c. can be expressed as

$$\left| \frac{d}{dt} \left(\tilde{f}''(t)^{-1/2} \right) \right| \leq 1. \tag{2.12}$$

By integrating (2.12) between 0 and $t \geq 0$, gives

$$\frac{\tilde{f}''(0)}{\left(1 + t\tilde{f}''(0)^{1/2}\right)^2} \leq \tilde{f}''(t) \leq \frac{\tilde{f}''(0)}{\left(1 - t\tilde{f}''(0)^{1/2}\right)^2}. \tag{2.13}$$

The upper bound is valid if $0 \leq t \leq \tilde{f}''(0)^{-1/2}$. By integrating twice the upper bound in (2.13), we obtain an upper bound for $\tilde{f}$

$$\begin{aligned} \tilde{f}(t) &\leq \tilde{f}(0) + t\tilde{f}'(0) - t\tilde{f}''(0)^{1/2} - \log\left(1 - t\tilde{f}''(0)^{1/2}\right) \\ &= \tilde{f}(0) - t\lambda(x)^2 - t\lambda(x) - \log\left(1 - t\lambda(x)\right). \end{aligned} \tag{2.14}$$

Where this bound on $\tilde{f}(t)$ is valid for $0 \leq t \leq 1/\lambda(x)$.

Let t_{bk} denote the first value of t which satisfies the condition of the backtracking line-search

$$\tilde{f}(\hat{t}) - \tilde{f}(0) \leq -\alpha\lambda^2\hat{t} \tag{2.15}$$

when an initial value of $t = 1$ is used, and consecutive multiplications of t by $\beta \in (0, 1)$ are performed until the condition is satisfied. From (2.14), it follows that $t_{\text{bk}} \geq \beta/(1 + \lambda)$ since at the point

$$\hat{t} = 1/\left(1 + \lambda(x)\right)$$

the line-search condition is satisfied, i.e.,

$$\begin{aligned}\tilde{f}(\hat{t}) - \tilde{f}(0) &\leq -\hat{t}\lambda^2 - \hat{t}\lambda - \log\left(1 - \hat{t}\lambda\right) \\ &= -\lambda + \log\left(1 + \lambda\right) \\ &\leq -\frac{1}{2}\frac{\lambda^2}{1+\lambda} \\ &\leq -\alpha\lambda^2\hat{t}\end{aligned}$$

where the second inequality holds since

$$\log\left(1 + x\right) \leq x - \frac{x^2}{2}\frac{1}{1+x}, \quad x > 0.$$

Since $t_{\text{bk}} \geq \beta/(1+\lambda)$ we have

$$\tilde{f}(t_{\text{bk}}) - \tilde{f}(0) \leq -\alpha\beta\frac{\lambda^2}{1+\lambda} \equiv -\gamma$$

so for the n-th iteration we have

$$\gamma^{(n)} = \alpha\beta\left(\frac{\lambda^{(n)2}}{1+\lambda^{(n)}}\right).$$

One can bound $\gamma^{(n)}$ for all the iterations of the damped Newton phase by lower bounding the Newton decrement $\lambda^{(n)}$ with a certain constant η which is later determined (i.e., $\lambda^{(n)} \geq \eta$). Thus the reduction in the value of f at each iteration during the damped convergence phase is at least

$$\gamma = \alpha\beta\frac{\eta^2}{1+\eta}. \tag{2.16}$$

This proves (2.9).

2.3.2 Quadratic convergence phase

Next, we show that we can use $\eta = (1-2\alpha)/4$ to upper bound λ in the quadratic convergence phase. Suppose $\lambda \leq (1-2\alpha)/2$, then we have using (2.14) that

$$\begin{aligned}\tilde{f}(1) - \tilde{f}(0) &\leq -\lambda^2 - \lambda - \log\left(1 - \lambda\right) \\ &\leq -\frac{1}{2}\lambda^2 + \lambda^3 \\ &\leq -\alpha\lambda^2.\end{aligned}$$

Therefore, the backtracking condition (2.15) is satisfied with $t = 1$ (i.e., a full step). The second inequality follows from the fact that $-x - \log(1-x) \leq \frac{1}{2}x^2 + x^3$ for $0 \leq x \leq 0.81$.

Assuming that $x^{(n+1)} = x^{(n)} - \nabla^2 f(x^{(n)})^{-1} \nabla f(x^{(n)})$, it is shown in [45] that if $\lambda(x^{(n)}) < 1$ and a full step is taken (i.e., $t = 1$), the following bound holds

$$\lambda(x^{(n+1)}) \leq \frac{\lambda(x^{(n)})^2}{(1 - \lambda(x^{(n)}))^2}. \tag{2.17}$$

Therefore, if $\lambda(x^{(n)}) \leq 1/4$, then $\lambda(x^{(n+1)}) \leq 2\lambda(x^{(n)})^2$ which yields that (2.10) holds when $\lambda(x^{(n)}) \leq \eta$. If we choose $\eta = (1 - 2\alpha)/4$, then $\lambda(x^{(n)}) \leq \eta \leq 1/4$ implies that $\lambda(x^{(n+1)}) \leq 1/4$. A substitution of η in (2.16) gives that

$$\gamma = \frac{\alpha\beta(1-2\alpha)^2}{20 - 8\alpha}. \tag{2.18}$$

2.3.3 Final complexity bound

Combining the results for the two phases, an upper bound on the number of Newton iterations gets the form

$$\begin{aligned} N &\leq \frac{f(x^{(0)}) - p^*}{\gamma} + \log_2 \log_2\big(\frac{1}{\varepsilon}\big) \\ &= \left(\frac{20 - 8\alpha}{\alpha\beta(1-2\alpha)^2}\right)\big(f(x^{(0)}) - p^*\big) + \log_2 \log_2\big(\frac{1}{\varepsilon}\big). \end{aligned} \tag{2.19}$$

This upper bound on the number of Newton iterations also holds for ECSCM problems since it satisfies the following lemma.

Lemma 1 Consider an ECSCM problem and it's degenerated UCSCM problem (same problem, excluding the constraints). Assume the problems are solved using Newton's method. Any bound on the number of Newton iterations or the Newton's decrement for the un-constrained problem also hold for the equality-constrained problem if the bound's dependence on the value of the objective function f and the equality constrains $Ax = b$ may be expressed in terms of $f(x^{(0)})$, p^* and $\lambda^{(n)}$. Where $f(x^{(0)})$ denote the value of f at the first iteration, p^* denote the optimal value and $\lambda^{(n)}$ denote the the Newton decrement at the n-th iteration.

Proof: As shown in [45, Chapter 10], an ECSCM problem with variable x

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \end{array}$$

can be reduced into an un-constrained problem with variable z

$$\text{minimize } \tilde{f}(z) = f(Fz + \hat{x})$$

where $AF = 0$ and $\hat{x}$ is any particular solution of $A\hat{x} = b$. This problem is shown to iterate the same as the original problem, having the same Newton's decrement, step direction and number of Newton iterations. The reduced problem is an un-constrained s.c. minimization problem, therefore we can use bounds for un-constrained s.c. minimization problem. Since the Newton's decrement, the number of Newton iterations, the minimum of the objective function and the value of the objective function at the first iteration are the same for the original and the reduced problem then the bound will hold for both problems.

We note that the bound in (2.19) depends only on the backtracking line-search parameters α and β, and the required tolerance ε in the minimization of the s.c. function f. Moreover, the term involving the repeated logarithm can be safely replaced by a rather small constant (e.g., 6), so the bound essentially depends only on α, β and the initial value $x^{(0)}$. For typical values of α and β, the constant that multiplies $f(x^{(0)}) - p^*$ is on the order of several hundreds. For example, for $\alpha = 0.1$ and $\beta = 0.8$, the corresponding scaling factor in (2.19) is equal to 375. With tolerance $\varepsilon = 10^{-10}$, we obtain the following upper bound on the number of iterations used by Newton's method for an s.c. function f:

$$375\big(f(x^{(0)}) - p^*\big) + 6.$$

For s.c. functions of the form $f = -\sum_{i=1}^{m} \log\big(b_i - a_i^T x\big)$, where the vectors $\underline{a}$ and $\underline{b}$ are randomly generated, simulated results indicate that the number of iterations scales like

$$c(f(x^{(0)}) - p^*) + 5 \tag{2.20}$$

where the value of c is nearly 1. Comparing these numerical results to the above upper bound, it follows that the bound is fairly conservative. However, it captures what appears to be the general form of the worst-case number of Newton steps required.

2.4 Tightened complexity bounds on Newton's method and interior-point method

In this section, the convergence behavior of Newton's method is further analyzed in order to derive tightened upper bounds on the number of Newton iterations. The tightening of the classical bound in Theorem 2 is done in two different approaches:

1. An analysis with pre-determined step size t (as opposed to the backtracking line-search).

2. A refined analysis of the original bound for the backtracking line-search.

As in Section 2.3, we derive upper bounds on the number of iterations during the damped and quadratic Newton phases, and then obtain an upper bound on the overall number of iterations by adding these two particular bounds, i.e.,

$$N_{\text{Total}} \leq N_{\text{Total}}^{\text{bound}} = N_{\text{Damped}}^{\text{bound}} + N_{\text{Quad}}^{\text{bound}}.$$

Where $N_{\text{Damped}}^{\text{bound}}$ and $N_{\text{Quad}}^{\text{bound}}$ designate the upper bound on the number of iterations during the damped and quadratic Newton phases, respectively.

The new bounds also scale like (2.20), but with a considerably smaller scaling factor c for the initial error $\big(f(x^{(0)}) - p^*\big)$. Next, these improved bounds are used to get an improved upper bound on the overall number of iterations that is required for an interior-point method in conjunction with Newton's method and logarithmic barriers.

The motivation for deriving these new bounds lays in its application in communication theory. The use of interior-point methods for decoding low-density parity-check codes on linear channels was recently introduced in [51]. In chapter 3, we apply these new tightened bounds to an IPM-based LP decoder in order to provide better estimates for the decoding complexity of such decoding algorithms.

2.4.1 Theorem 4 - Complexity bound for ECSCM problems with pre-determined step size

In this subsection and in the following one, we focus on tightening the classical analysis using a pre-determined step size t, as opposed to the backtracking line-search. These

tightened bounds reduce the scaling factor for the initial error $\big(f(x^{(0)}) - p^*\big)$ by factor of 10 to 100.

Theorem 4 [Parametric complexity bound for ECSCM problem solved using Newton's method with a pre-determined step size t]
Consider an ECSCM problem solved using Newton's method with the following pre-determined choice of $t^{(n)}$:

$$t^{(n)} = \begin{cases} \frac{1}{1+\lambda^{(n)}} & \lambda^{(n)} \geq \eta \quad \text{Damped convergence phase} \\ 1 & \lambda^{(n)} < \eta \quad \text{Quadratic convergence phase} \end{cases}$$

where $0 < \eta < \frac{3-\sqrt{5}}{2} \approx 0.3819$ is a free parameter. Let $x^{(0)}$ be the initial point used for Newton's method. Then the number of iterations required to find p^* with accuracy ε is upper bounded by

$$N_{\text{Total}} \leq c_1\big(f(x^{(0)}) - p^*\big) + c_2 \tag{2.21}$$

where

$$\begin{aligned} c_1 &= \frac{1}{\eta - \log(1+\eta)} \\ c_2 &= \left\lceil \log_2 \left(\frac{\log_2(\sqrt{\varepsilon}/(1-\eta)^2)}{\log_2(\eta/(1-\eta)^2)} \right) \right\rceil . \end{aligned} \tag{2.22}$$

Proof: See Section 2.5.1

By changing the bound parameter $\eta \in (0, 0.381...)$, one obtains a tradeoff between the two terms in (2.21) as is evidenced in Table 2.1 and Fig. 2.4.

While increasing the value of η towards 0.381, the scaling factor of the initial error, c_1, is decreased whereas the additive constant c_2 is increased and tends to infinity.

In Fig. 2.5, the parameter η which optimizes the bound is plotted versus the initial error $f(x^{(0)}) - p^*$. It can be seen that for most of the values of $f(x^{(0)}) - p^*$, the optimal value for η is between 0.378 to 0.38 (practically this graph is meaningful only after applying the normalization scaling described in remark 5). This result is very narrow and close to $\eta^{\max}$. This indicates that c_1 is dominant compared to c_2. Thus the use of large values of η (e.g., $\eta \cong 0.378$) are preferable.

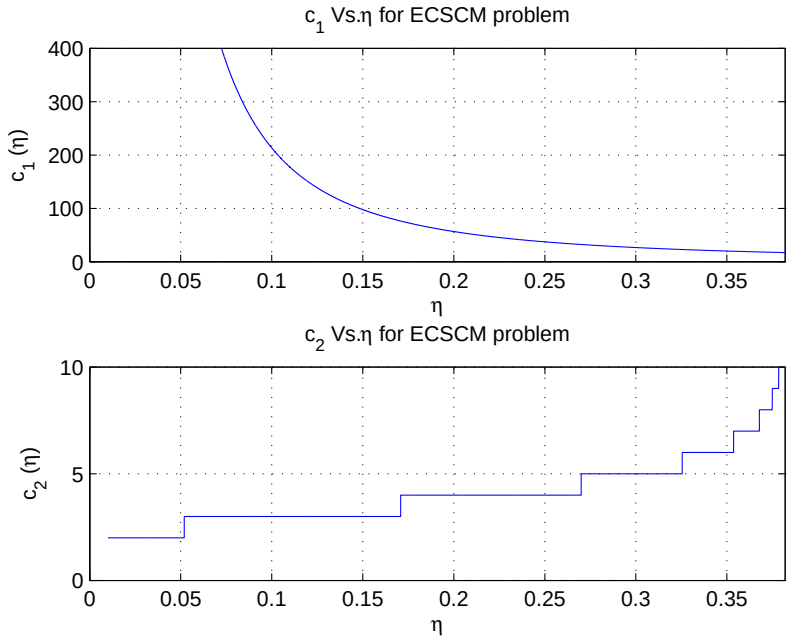

Figure 2.4: The coefficients c_1 and c_2 in the upper bound on the number of iterations, as given in (2.21) and (2.22) for some values of the free parameter $\eta \in (0, 0.381)$. The additive constant c_2 in (2.21) refers to $\varepsilon = 10^{-10}$.

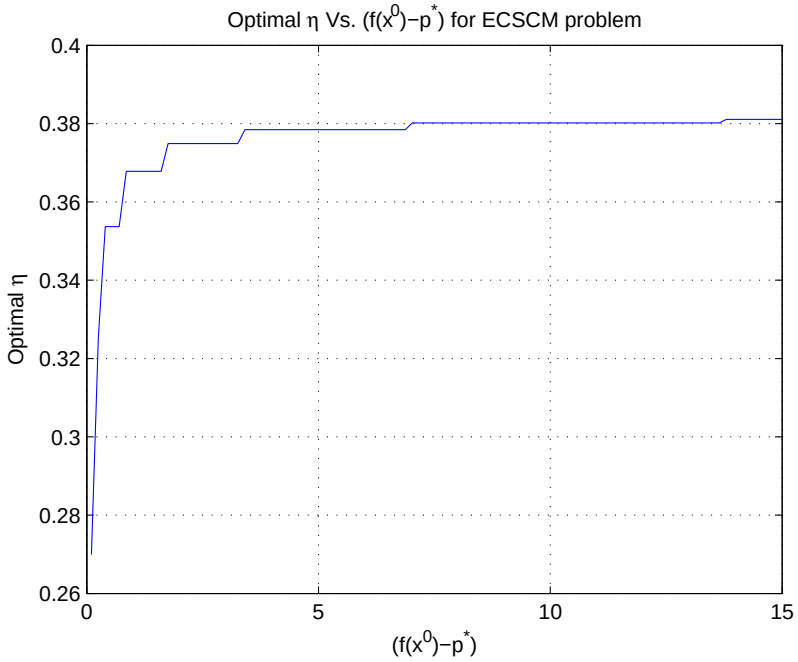

Figure 2.5: The optimal value of η versus the initial error $f(x^{(0)}) - p^*$ when the bound in Theorem 4 is considered.

η	c_1	c_2
0.250	37.2	5
0.289	28.3	5
0.322	33.1	6
0.353	19.7	7
0.378	17.3	11
$\frac{3-\sqrt{5}}{2}$	17.1	∞

Table 2.1: The coefficients c_1 and c_2 in the upper bound on the number of iterations, as given in Theorem 4.

Remark 4 We note that a predetermined choice of t is a suboptimal choice compared to exact line-search. Thus, the upper bounds derived also apply to exact line-search.

Remark 5 Scaling of the objective function $f(x)$ and the desired tolerance ε by a factor $k > 1$ results in an equivalent problem with identical number of Newton iterations. However, the number of Newton iterations in the bound above scales linearly with $f(x) - p^*$ rather then being invariant to scaling in $f(x)$. This can be resolved by applying a normalization scaling to $f(x)$ (in a manner that conserves the s.c. property) before applying the bound. For example, consider a function $f(x)$ that fulfills the inequality

$$|f^{(3)}(x)| \leq cf''(x)^{3/2}.$$

Multiplying the function by the constant $k = \frac{c^2}{4}$ results in the function $\tilde{f}(x) = kf(x)$ which complies with the s.c. inequality (6) with equality (see [45, p. 498]). In addition a tighter bound is now obtained, since applying the bound on the normalized version (i.e., $\tilde{f}(x) = kf(x)$ and $\tilde{\varepsilon} = k\varepsilon$) of the original problem will tighten the first term of the bound by a factor k with minor effect (if any) on the additive term.

2.4.2 Theorem 5 - Complexity bound for ECSCM problems with an improved pre-determined step size

Theorem 5 [Parametric complexity bound for ECSCM problem solved using Newton's method with an Improved pre-determined step size t **]** Consider an ECSCM problem solved using Newton's method. Let $\eta \in (0, 1)$ be chosen arbitrarily, and consider the following pre-determined choice of the step size $t^{(n)}$:

- If $\lambda^{(n)} \geq \eta$, then $t^{(n)} = \frac{1}{1+\lambda^{(n)}}$.
- Otherwise, if $\lambda^{(n)} < \eta$, let

$$t^{(n)} = \arg\min_{t\in(0,1]} c(t, \lambda^{(n)}) \tag{2.23}$$

where

$$\begin{aligned} a(t,\lambda) &= 1 - \frac{t}{1-\lambda t} \\ b(t,\lambda) &= 1 - \frac{1-\big(1-\lambda t\big)^3}{3\lambda} \\ c(t,\lambda) &= \frac{\max(|a(t,\lambda)|, |b(t,\lambda)|)}{1-\lambda t}. \end{aligned} \tag{2.24}$$

Then the number of iterations required to find p^* with accuracy ε is upper bounded by

$$N_{\text{Total}} \leq c_1\big(f(x^{(0)}) - p^*\big) + \tilde{c}_2$$

where c_1 is given in (2.22) for $\eta \in (0,1)$, and the additive term $\tilde{c}_2$ (which is not expressed in closed-form but calculated numerically) has the feature that it tends to infinity as we let $\eta \to 1$.

Proof: See Section 2.5.2.

By changing the parameter η, we perform a tradeoff between c_1 and $\tilde{c}_2$ as shown in Table 2.2.

We note that with the improved choice of t, the parameter η can be chosen close to one thus tightening the term c_1 considerably. This property of the additive term $\tilde{c}_2$ as above is in contrast to c_2 in Theorem 4 which tends to infinity as we let η tend to 0.381 from below.

2.4.3 Theorem 6 - Complexity bound for ECSCM problems with backtracking line-search

In this subsection, we provide a bound on the number of Newton iterations when solving an ECSCM problem using Newton's method and backtracking line-search.

η	c_1	$\tilde{c}_2$
0.250	37.25	4
0.381	17.18	5
0.700	5.90	10
0.900	3.87	35
0.990	3.31	392
1.000	3.26	∞

Table 2.2: The coefficients c_1 and $\tilde{c}_2$ in the upper bound of Theorem 5 for some values of the free parameter $\eta \in (0,1)$. The calculation of the additive constant $\tilde{c}_2$ is given for an accuracy of $\varepsilon = 10^{-10}$ in the calculation of p^*.

This Theorem is an improved version of the classical result given in Theorem 2. The new Theorem tighten the scaling factor of $f(x) - p^*$ by factor 10-100 (Depending on the backtracking line-search parameters) as compared to Theorem 2 .

Theorem 6 [Improved complexity bound for ECSCM problems solved with Newton's method and backtracking line-search]

Consider an ECSCM problem solved using Newton's method and backtracking line-search. Let $\alpha \in (0, 1/2)$ and $\beta \in (0,1)$ be the parameters of the backtracking line-search. Let $\eta_{\max} \in (0, \frac{3-\sqrt{5}}{2})$ be a free parameter and define $\eta \equiv \min\left(\frac{1}{2}\frac{1-2\alpha}{1-\alpha}, \eta_{\max}\right)$.

Then the number of Newton iterations is upper bounded by

$$N_{\text{total}} \leq N_{\text{Damped}}^{\text{bound}} + N_{\text{Quad}}^{\text{bound}} \tag{2.25}$$

where

$$N_{\text{Damped}}^{\text{bound}} = \frac{f\left(x^{(0)}\right) - p^*}{\alpha\beta\eta^2}$$
$$N_{\text{Quad}}^{\text{bound}} = \left\lceil \log_2\left(\frac{\log_2(\sqrt{\varepsilon}/(1-\eta)^2)}{\log_2(\eta/(1-\eta)^2)}\right)\right\rceil.$$

Proof: See Section 2.5.3.

The parameter $\eta_{\max}$ is used to make $N_{\text{Quad}}^{\text{bound}}$ be finite. If η was not bounded by $\eta_{\max}$, then for $\alpha \leq \frac{3-\sqrt{5}}{4} \approx 0.19$ the value of η was larger then $\frac{3-\sqrt{5}}{2}$, this would cause the bound for N_{Quad} to diverge to infinity.

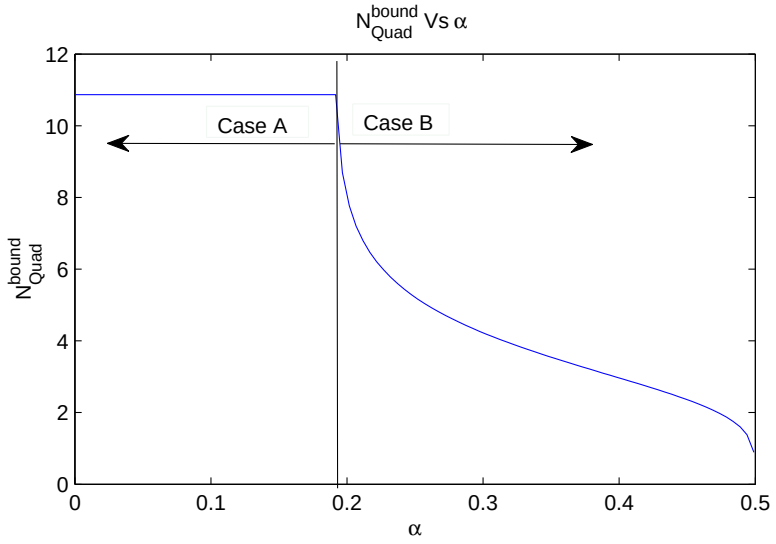

Figure 2.6: The term $N_{\text{Quad}}^{\text{bound}}$ in (2.25) versus α for tolerance $\varepsilon = 10^{-10}$. The behavior of the bound can be classified into two cases depending on the value of α.

As seen in Figures 2.6 and 2.7, the bound above can be classified into two cases depending on the value of α.

- Case A -$\alpha < \frac{1}{2}\frac{1-2\eta_{\max}}{1-\eta_{\max}}$ The value of η is bounded by $\eta_{\max}$ in order to limit the value of $N_{\text{Quad}}^{\text{bound}}$. In this case $N_{\text{Quad}}^{\text{bound}}$ is invariant to changes in α. The value of $N_{\text{Damped}}^{\text{bound}}$ decreases as we increase the value of α.

- Case B -$\alpha \geq \frac{1}{2}\frac{1-2\eta_{\max}}{1-\eta_{\max}}$ The parameter $\eta_{\max}$ does not affect the bound. In this case $\eta = \frac{1}{2}\frac{1-2\alpha}{1-\alpha}$ and the value of $N_{\text{Damped}}^{\text{bound}}$ increases as we increase the value of α.

Figure 2.7 plots the term for the damped phase in the improved bound, as compared to the original bound. The improved bound receives its minima of $\frac{N_{\text{Damped}}^{\text{bound}}}{f(x^{(0)})-p^*} = \left(\frac{\sqrt{17}-1}{\sqrt{17}-3}\right)^2\left(\frac{4}{5-\sqrt{17}}\right) \cong 35.27$ at $\alpha = \frac{5-\sqrt{17}}{4} \cong 0.21$. The original bound also shows a minima around $\alpha \approx 0.2$, however it is almost 10 times looser compared to the improved bound. If we let $\alpha \to 1/2$ or $\alpha \to 0$ the original bounds loosens much faster then the improved bound. We note that the dependence of the improved bound on α is very flat around its minima. This weak dependence on the value of α coincides with simulated results that show small changes in the number of Newton iterations

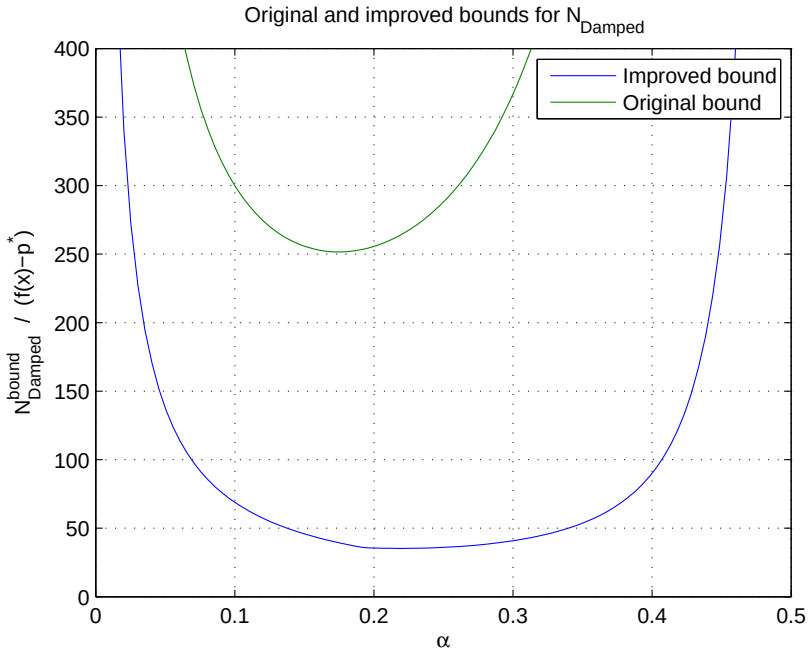

Figure 2.7: Improved (2.25) and original (2.19) bound for N_{Damped} versus α. The bounds are scaled by factor $f(x^{(0)}) - p^*$ and shown for $\beta = 1$ and $\eta_{\max} = 0.38$.

when changing α.

2.4.4 Theorem 7 - Bounds' extension to equality and inequality constrained minimization using interior-point methods

In this subsection, we extend the results given in Theorems 4, 5 and 6 for an ECSCM problem to the case of an IECSCM problem (i.e., problem with inequality constraints). The theorem assumes that the problem is solved using interior-point methods with logarithmic barrier.

Theorem 7 [Bounds' extension to equality and inequality constrained minimization using interior-point methods]
Consider an IECSCM optimization problem. Assuming the problem is solved using an interior-point method (IPM) with logarithmic barrier (where the parameters of the outer iterations are set to $t^{(0)}, \mu$, and the inner iterations are performed using Newton's method), then the number of Newton iterations is upper bounded by

$$N_{\text{Total}}^{\text{Inequality}} = N_{\text{outer}} N_{\text{inner}} + N_{\text{initial}} \leq \left\lceil \frac{\log\left(m/(\varepsilon t^{(0)})\right)}{\log \mu} \right\rceil \left(\frac{m\left(\mu - 1 - \log \mu\right)}{\gamma} + c \right) + N_{\text{initial}}$$

The numbers γ and c are calculated by excluding the inequality constraints in the original problem and applying the appropriate bound in Theorems 4, 5 or 6 depending on the method t is calculated. The numbers γ and c are extracted by comparing the bound to the form "$N_{\text{Total}} \leq \frac{f(x^{(0)}) - p^*}{\gamma} + c$"

Proof: See Section 2.5.4.

Remark 6 We note that the bound on N_{inner} is independent of t, suggesting that the number of inner iterations does not change much from one outer iteration to the next.

2.5 Proofs of Theorems

In this section, we provide proofs to the theorems introduced in Section 2.4. The proofs of Theorems 4, 5 and 6 are given for UCSCM problem (disregarding the equality

constraints of the original problem). Using Lemma 1, the proofs can be extended to an ECSCM problem without any change.

2.5.1 Proof of Theorem 4

The proof consists of an analysis of the damped and quadratic convergence phases of the Newton method.

Damped convergence phase

Using (2.14), a bound on the reduction in the value of f at the n-th iteration is obtained

$$\Delta f(t^{(n)}) \equiv f(x^{(n)}) - f(x^{(n+1)}) = \tilde{f}(0) - \tilde{f}(t^{(n)}) \geq t^{(n)} \left(\lambda^{(n)}\right)^2 + t^{(n)}\lambda^{(n)} + \log\left(1 - t^{(n)}\lambda^{(n)}\right).$$

The maximization of the lower bound on $\Delta f(t^{(n)})$ is obtained by nulling the derivative with respect to $t^{(n)}$. This gives the equality

$$\left(\lambda^{(n)}\right)^2 + \lambda^{(n)} - \frac{\lambda^{(n)}}{1 - t^{(n)}\lambda^{(n)}} = 0 \Rightarrow t^{(n)} = \frac{1}{1 + \lambda^{(n)}}.$$

Therefore

$$\Delta f\left(t^{(n)}\right) \geq \lambda^{(n)} - \log\left(1 + \lambda^{(n)}\right). \tag{2.26}$$

This justifies the choice $t^{(n)} = \frac{1}{1+\lambda^{(n)}}$ as a predetermined value of $t^{(n)}$. Since an explicit expression of $\lambda^{(n)}$ for any iteration, in terms of the parameters of the Newton's method and the function f, is not available, then (similarly to the classic analysis) we lower bound $\lambda^{(n)}$ in the damped convergence phase by a constant η which is later determined. Since (2.26) is monotonically increasing with λ then

$$\Delta f\left(t^{(n)}\right) \geq \min_{\lambda \geq \eta} \left(\lambda^{(n)} - \log\left(1 + \lambda^{(n)}\right)\right) = \eta - \log\left(1 + \eta\right).$$

Hence, the number of Newton iterations during the damped phase is at most

$$N_{\text{damped}} \leq \frac{f\left(x^{(0)}\right) - p^*}{\eta - \log\left(1 + \eta\right)}.$$

This proves the expression for c_1 in (2.22).

Quadratic convergence phase

We now show that once $\lambda \leq \eta$, a choice of $t = 1$ yields a double exponential decay of λ which results in a very fast convergence to the minimal value of f (denoted by p^*). It is shown in [45] that if we choose $t = 1$, the following bound holds

$$\lambda(x^{(n+1)}) \leq \frac{\lambda(x^{(n)})^2}{(1-\lambda(x^{(n)}))^2} \quad for\ \lambda(x^{(n)}) < 1 \tag{2.27}$$

Using the bound in (2.27), we can ensure a decay if $\lambda(x^{(n+1)}) \leq c\lambda(x^{(n)})$ where $c < 1$. By taking the limit $c \to 1$, we get $\frac{\lambda^2}{(1-\lambda)^2} = \lambda \Rightarrow \eta^{\max} = \frac{3-\sqrt{5}}{2} \cong 0.381$. Increasing the value of η tightens the bound on the number of iterations during the damped phase. However, in the vicinity of 0.381, the bound for the quadratic phase becomes useless because it doesn't converge.

Denote $a \doteq \frac{1}{(1-\eta)^2}$ where $\eta \in (0, 0.381)$. If $\lambda(x) \leq \eta$, then using (2.27) we get that $\lambda(x^{(n+1)}) \leq \frac{\lambda(x^{(n)})^2}{(1-\eta)^2} \triangleq a\lambda(x^{(n)})^2$. Applying this inequality recursively gives

$$a\lambda\big(x^{(l)}\big) \leq \big(a\lambda\big(x^{(k)}\big)\big)^{2^{l-k}}, \quad l \geq k. \tag{2.28}$$

At the final iteration λ is very small (in the order of $\sqrt{\varepsilon}$), thus we may use the following inequality (see [45, Eq. (9.49)]):

$$f(x^{(l)}) - p^* \leq \lambda(x^{(l)})^2, \quad \forall\, \lambda \leq 0.68. \tag{2.29}$$

Combining (2.28) and (2.29) gives

$$\sqrt{\varepsilon} = \sqrt{f(x^{(l)}) - p^*} \leq \lambda(x^{(l)}) \leq \frac{\big(a\lambda(x^{(k)})\big)^{2^{l-k}}}{a} \underset{a\lambda \leq a\eta}{\leq} \frac{(a\eta)^{2^{l-k}}}{a} \tag{2.30}$$

An upper bound on the number of iterations during the quadratic phase is obtained by solving the inequality for $N_{\text{quad}} = l - k$, which gives that

$$N_{\text{quad}} = l - k \leq \log_2 \left(\frac{\log_2 (a\sqrt{\varepsilon})}{\log_2 (a\eta)} \right). \tag{2.31}$$

Ceiling this expression and substituting $a = \frac{1}{1-\eta^2}$ proves the term for c_2 in (4).

2.5.2 Proof of Theorem 5

Similarly to the classical analysis, we address in the following the two convergence phases of the Newton method.

Damped convergence phase

As before $t^{(n)} = \frac{1}{1+\lambda^{(n)}}$ maximizes the bound for $\Delta f\left(t^{(n)^*}\right)$ which results in $\Delta f\left(t^{(n)^*}\right) \leq \lambda^{(n)} - \log\left(1+\lambda^{(n)}\right)$. Using $\lambda^{(n)} \geq \eta$ for the damped phase we get $\gamma^{(n)} \leq \gamma \leq \eta - \log(1+\eta)$

Quadratic convergence phase

The proof relies on the following lemma:

Lemma 2 *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a strictly convex s.c. function. Suppose that $\lambda(x)t < 1$ and define $x^+ = x + t\left(-\nabla^2 f(x)^{-1}\nabla f(x)\right) = x + t\Delta x_{nt}$. Then the following inequality holds*

$$\frac{\lambda(x^+)}{\lambda(x)} \leq c(t, \lambda(x)) \quad \lambda(x)t \leq 1. \tag{2.32}$$

where

$$\begin{aligned} a(t,\lambda) &= 1 - \frac{t}{1-\lambda t} \\ b(t,\lambda) &= 1 - \frac{1-\left(1-\lambda t\right)^3}{3\lambda} \\ c(t,\lambda) &= \frac{\max(|a(t,\lambda)|, |b(t,\lambda)|)}{1-\lambda t}. \end{aligned} \tag{2.33}$$

Proof: See proof of Lemma 2 in Appendix A.

Since minimizing $c(t, \lambda(x))$ maximizes the decay rate, the best predetermined choice for $t^{(n)}$ using this bound is to use $t^{(n)} = t_{\text{pre}}(\lambda) = \arg\min_{t\in(0,1]} c(t,\lambda)$. Figure 2.8 plots the decay rate $c(t_{\text{pre}}, \lambda)$ and t_{pre} Versus λ, as given in (2.24) and (2.23) respectively. The figure shows that the condition for decay, $c(t_{\text{pre}}, \lambda) < 1$, holds for $\lambda < 1$. Therefore the improved choice of t_{pre} results in a bound that converges for values of η approaching one.

In order to compute the bound on the number of Newton iterations in the quadratic phase using this choice of $t^{(n)}$, one needs to count numerically the number of iterations required to apply inequality (2.32) recursively starting from $\lambda = \eta$ until $\lambda \leq \sqrt{\varepsilon}$. This

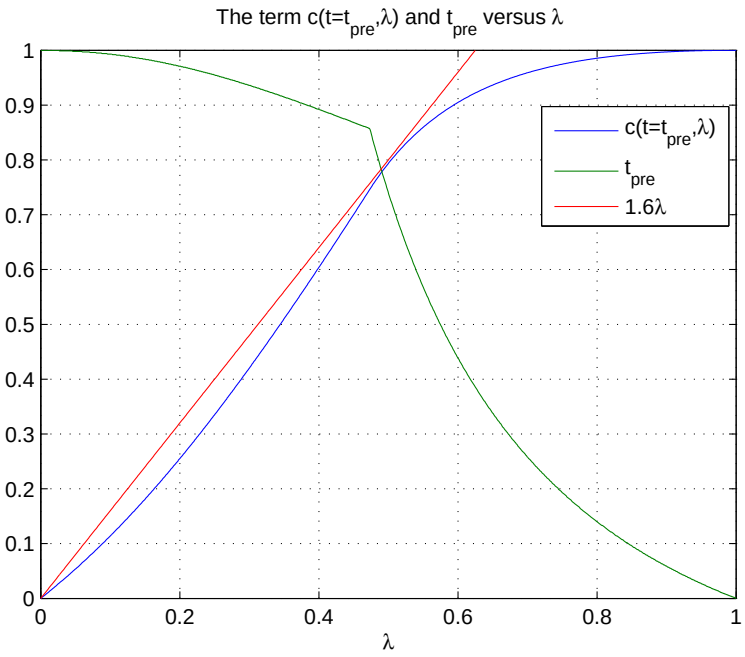

Figure 2.8: The function $c(t = t_{\text{pre}}, \lambda)$ and t_{pre} Versus λ, as given in (2.24) and (2.23) respectively. The line 1.6λ is also plotted to show it upper bounds the function $c(t = t_{\text{pre}}, \lambda)$.

method can be simplified by performing most of the counting analytically. Figure 2.8 shows that the quadratic phase has two convergence sub-phases:

1. Double exponential decay (i.e., $\lambda \leq 0.55$) - For small values of λ , $t_{\text{pre}} \approx 1$ and $c(t_{\text{pre}}, \lambda)$ is tightly upper bounded by the line 1.6λ. Therefore $\lambda(x^{(n+1)}) \leq 1.6\lambda(x^{(n)})^2$ and as in the proof given in Theorem 4, we have a double exponential decay. We can use (2.30) with $a = 1.6$ to bound the number of iterations in this sub-phase

$$N_{\text{Double exponential}}\,(a = 1.6) \leq \left\lceil \log_2 \left(\frac{\log_2 \left(1.6\sqrt{\varepsilon}\right)}{\log_2 \left(1.6 \min(\eta, 0.55)\right)} \right) \right\rceil . \tag{2.34}$$

   I.e., for $\eta > 0.55$ and $\varepsilon = 10^{-10}$ the bound is $N_{\text{Double exponential}} \leq 7$.

2. Slow exponential decay - For higher values of λ, t_{pre} approaches zero and $c(t_{\text{pre}}, \lambda)$ is rather constant (but approaching 1). Therefore the decay rate is much slower. This can be seen as a transition phase between the damped phase and the double exponential phase. The bound on the number of iterations in this sub-phase has to be calculated numerically by applying (2.32) recursively starting from $\lambda = \eta$ until $\lambda = 0.55$. The numerical result for this calculation is given in Figure 2.9

The term $\tilde{c}_2$ is the sum of the results from the two sub-phases $\tilde{c}_2 = N_{\text{Double exponential}} + N_{\text{slow exponential}}$

2.5.3 Proof of Theorem 6

Similarly to previous proofs, we address in the following the two convergence phases of the Newton method.

Damped convergence phase

Let $t_{\text{bk}} \in (0, 1]$ denote the first value of t which satisfies the backtracking line-search exit condition (2.15), where an initial value of $t = 1$ is used and consecutive multiplications of t by $\beta \in (0, 1)$ are performed until condition (2.15) is satisfied. Since the backtracking condition is satisfied at $t = t_{\text{bk}}$, then $\Delta f(t^{(n)})$ is lower bounded by

$$\Delta f(t^{(n)}) \equiv f(x^{(n)}) - f(x^{(n+1)}) = \tilde{f}(0) - \tilde{f}(t^{(n)}) \geq \alpha\lambda^2 t_{\text{bk}}^{(n)}. \tag{2.35}$$

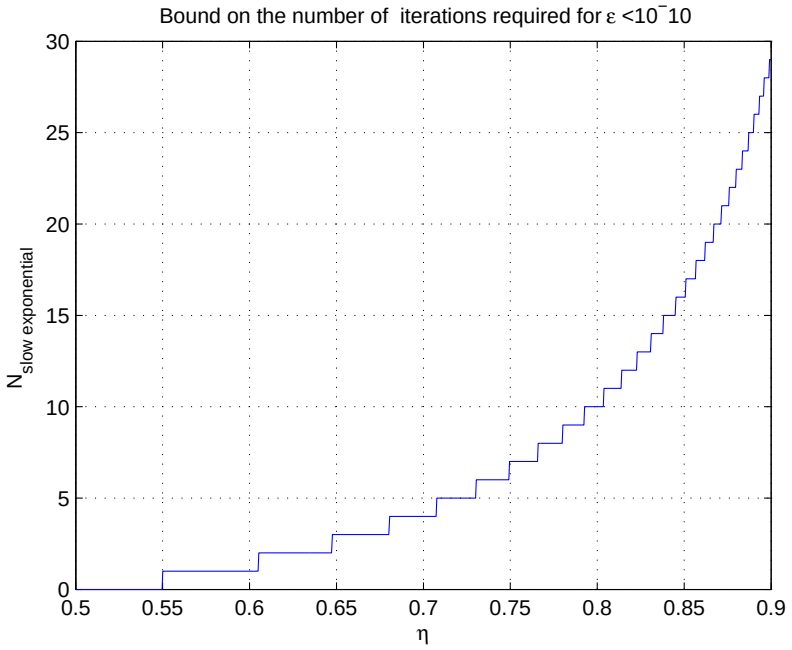

Figure 2.9: Bound on the number of iterations in the slow exponential decay sub phase vs η. The bound on the number of iterations in this sub-phase has to be calculated numerically by applying (2.32) recursively starting from $\lambda = \eta$ until $\lambda = 0.55$.

In order to lower bound the value of t_{bk} we shall use the following lemma:

Lemma 3 Let f be a s.c function minimized using Newton's method and backtracking-line-search. Let $t_{\text{exit}} \in (0, 1]$ denote the largest value of t which satisfies the backtracking exit condition in (2.15). Then t_{exit} is lower bounded by

$$t_{\text{exit}} \geq \min\left(1, \frac{G(\alpha)}{1 + \lambda G(\alpha)}\right) \tag{2.36}$$

Where $G(\alpha) \equiv 2(1 - \alpha)$ and λ denotes the Newton decrement.

Proof: First, we show that $\hat{t} = \frac{G(\alpha)}{1+\lambda G(\alpha)}$ satisfies the backtracking exit condition in (2.15).

$$\begin{aligned}
\tilde{f}(\hat{t}) - \tilde{f}(0) &\overset{(a)}{\leq} -\hat{t}\lambda^2 - \hat{t}\lambda - \log(1 - \hat{t}\lambda) \\
&\overset{(b)}{=} -\hat{t}\lambda^2 - \hat{t}\lambda - \log\left(1 + \frac{\hat{t}\lambda}{1 - \hat{t}\lambda}\right) \\
&\overset{(c)}{\leq} \hat{t}\lambda\left[-\lambda - 1 + \frac{1 - \hat{t}\lambda/2}{1 - \hat{t}\lambda}\right] \\
&\overset{(d)}{=} -\alpha\lambda^2\hat{t}
\end{aligned}$$

where (a) follows from (2.14), (b) holds using the equality $-\log(1-x) = \log\left(1 + \frac{x}{1-x}\right)$ with $x = \lambda\hat{t}$, (c) follows from the inequality $\log(1 + x) \leq x - \frac{x^2}{2}\frac{1}{1+x}$ (Valid for $x \geq 0$) with $x = \frac{\hat{t}\lambda}{1-\hat{t}\lambda}$ and (d) follows by substitution $\hat{t} = \frac{G(\alpha)}{1+\lambda G(\alpha)}$. Hence $\hat{t}$ satisfies the exit condition in (2.15).

Define $g(t) \equiv \tilde{f}(0) - \tilde{f}(t) - \alpha\lambda^2 t$. Since $g(0) = 0$, $g(\hat{t}) \geq 0$ and $g(t)$ is a concave function (as a sum of two concave functions), then if $t \in [0, \hat{t}]$, then $g(t) \geq 0$ which implies that t satisfies the backtracking exit condition. Therefore if $\hat{t} \geq 1$ then the exit condition is satisfied for $t = 1$ as well. We conclude that $\min(1, \hat{t}) \in (0, 1]$ satisfies the exit condition and therefore it is a lower bound on t_{exit}.

Since t_{exit} is the largest value of t that satisfies the backtracking line-search exit condition, and the condition is checked only at values of the form $t = \beta^k$ ($k \in \mathbb{Z}^+$), then $t_{\text{bk}} = \beta^{\lceil \log_\beta(t_{\text{exit}}) \rceil}$. This result can be lower bounded as follows

$$t_{\text{bk}} = \beta^{\lceil \log_\beta(t_{\text{exit}}) \rceil} \geq \beta^{1+\log_\beta(t_{\text{exit}})} = \beta t_{\text{exit}}. \tag{2.37}$$

Using equations (2.35), (2.37), Lemma 3 and $\lambda^{(n)} \geq \eta \equiv \min\left(\frac{1}{2}\frac{1-2\alpha}{1-\alpha}, 1-\frac{1}{\sqrt{a_{\max}}}\right)$ we get

$$\begin{aligned}
\tilde{f}(0) - \tilde{f}(t_{\text{bk}}^{(n)}) &\geq \alpha\left(\lambda^{(n)}\right)^2 t_{\text{bk}}^{(n)} \\
&\geq \alpha\left(\lambda^{(n)}\right)^2 \beta t_{\text{exit}}^{(n)} \\
&\geq \alpha\beta\left(\lambda^{(n)}\right)^2 \min\left(1, \frac{G(\alpha)}{1+\lambda^{(n)}G(\alpha)}\right) \\
&\geq \alpha\beta\eta^2 \min\left(1, \frac{G(\alpha)}{1+\eta G(\alpha)}\right).
\end{aligned}$$

However, since $\eta \leq \frac{1}{2}\frac{1-2\alpha}{1-\alpha}$ then $\frac{G(\alpha)}{1+\eta G(\alpha)} \leq 1$ and $\min\left(1, \frac{G(\alpha)}{1+\eta G(\alpha)}\right) = 1$. Therefore $f\left(x^{(k+1)}\right) - f\left(x^{(k)}\right) \leq -\gamma \equiv -\alpha\beta\eta^2$ which means that the number of iterations in the damped phase is upper bounded by

$$N_{\text{Damped}} \leq \frac{f(x^{(0)}) - p^*}{\alpha\beta\eta^2}. \tag{2.38}$$

Quadratic convergence phase

Since $\lambda \leq \eta \leq \frac{1}{2}\frac{1-2\alpha}{1-\alpha}$ then inequality (2.36) becomes $t_{\text{exit}} \geq 1$. This means that a full Newton step is taken (i.e., $t = 1$). Since in addition $\lambda \leq \eta < 1$, then inequality (2.27) holds. As shown in the proof of Theorem 4, it follows that the number of Newton iterations during the quadratic phase satisfies the bound in (2.31). This bound diverges to infinity as $\eta \to \frac{3-\sqrt{5}}{2}$. Therefore, in order to make the bound finite, the expression for η contains a minimization with $\eta_{\max} \in (0, \frac{3-\sqrt{5}}{2})$. Combining this bound with (2.38) results in a bound on the total number of Newton iterations as argued in (2.25).

Remark 7 As shown in Figure 2.10, by observing the Newton decrement λ, we can trace the appearance of three phenomena concerning the convergence of the Newton method:

1. $\lambda \leq \lambda_a \equiv \frac{1}{2}\frac{1-2\alpha}{1-\alpha}$: From this point on, a full Newton step (i.e., $t = 1$) is assured.
2. $\lambda \leq \eta_{\max}$: From this point on, a double exponential decay (i.e., $\lambda^{(n+1)} \leq \left(\frac{\lambda^{(n)}}{1-\eta_{\max}}\right)^2$) is assured.

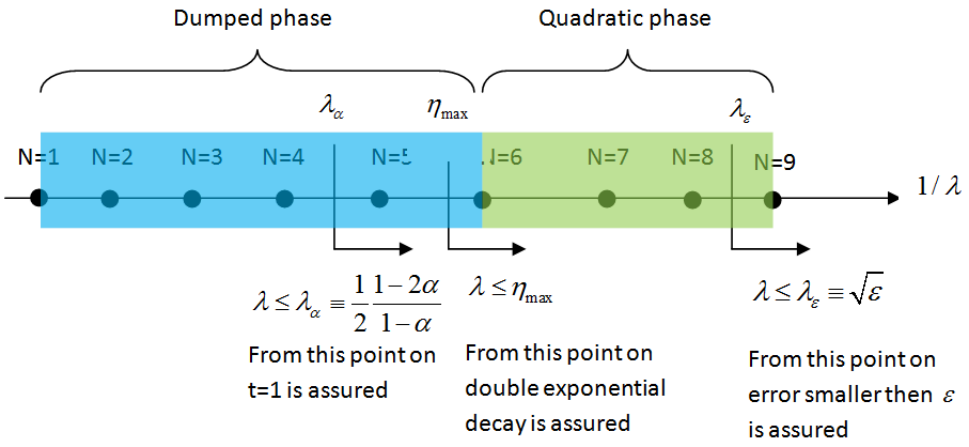

Figure 2.10: Consider Newton's method with backtracking line-search. By observing the Newton decrement λ, we can trace the appearance of three phenomena concerning the convergence of Newton's method . From a certain value (and below), it is assured that the backtracking line-search chooses a full step. Similarly, below the value of $\eta_{\max}$, a double exponential decay is assured. Next, below $\sqrt{\epsilon}$ it is assured that the current iterate is at most ϵ sub-optimal compared to p^*.

3. $\lambda \le \lambda_\varepsilon \equiv \sqrt{\varepsilon}$: From this point on, tolerance smaller then ε is assured.

The transition between the damped to the quadratic convergence phases occurs when the first two conditions are satisfied.

2.5.4 Proof of Theorem 7

The following proof is similar to a proof given in [45]. It is repeated to show that it still holds as an extension for the newly derived bounds.

Consider an optimization problem as described in Theorem 7. Solving the problem using interior-point methods (IPM) requires solving a series of un-constrained problems with parameter t :

$$\begin{aligned} &\text{minimize} && tf(x) + B(x) \\ &\text{subject to} && Ax = b \end{aligned} \tag{2.39}$$

where

$$B(x) = \sum_{i=1}^{m} -\log\left(-f_i(x)\right).$$

It can be shown that the solution to this problem $x^*(t)$ is m/t sub-optimal. Therefore, it becomes more accurate as t grows. If the parameter t is increased by factor μ at each outer iteration, then, the desired tolerance ε is achieved after exactly

$$N_{\text{outer}} = \left\lceil \frac{\log\left(m/\varepsilon t^{(0)}\right)}{\log \mu} \right\rceil \tag{2.40}$$

centering steps, plus the initial centering step. Each outer iteration involves solving an equality-constrained optimization problem with objective function $\tilde{f}(x) = \mu t \cdot f(x) + B(x)$. As a starting point, we use the solution from the previous outer iteration $x^*(t)$.

To lighten the notation, we use x to denote $x^*(t)$ at the current outer iteration, and x^+ to denote $x^*(\mu t)$ at the next iteration. We use λ and ν to denote $\lambda^*(t)$ and $\nu^*(t)$, respectively. The self-concordance assumption implies that the number of Newton inner iterations required to compute $x^+ \equiv x^*(\mu t)$ starting at $x \equiv x^*(t)$ is bounded by

$$N_{\text{inner}} \leq \frac{\tilde{f}(x) - \tilde{f}(x^+)}{\gamma} + c = \frac{\mu t f_0(x) + B(x) - \mu t f_0(x^+) - B(x^+)}{\gamma} + c \tag{2.41}$$

where γ and c are derived from the bound for the ECSCM problems which result by excluding the inequality constraints. We can bound this expression in terms of m and μ.

$$
\begin{aligned}
&\mu t f_0(x) + B(x) - \mu t f_0(x^+) - B(x^+) \\
&\overset{(a)}{=} \mu t f_0(x) - \mu t f_0(x^+) + \sum_{i=1}^{m} \log\left(-f_i(x^+)\right) - \sum_{i=1}^{m} \log\left(-f_i(x)\right) \\
&\overset{(b)}{=} \mu t f_0(x) - \mu t f_0(x^+) + \sum_{i=1}^{m} \log\left(-f_i(x^+)\right) - \sum_{i=1}^{m} \log\left(1/\left(\lambda_i t\right)\right) \\
&\overset{(c)}{=} \mu t f_0(x) - \mu t f_0(x^+) + \sum_{i=1}^{m} \log\left(-\mu\lambda_i t f_i(x^+)\right) - m\log\mu \\
&\overset{(d)}{\leq} \mu t f_0(x) - \mu t f_0(x^+) + \mu t \sum_{i=1}^{m} \lambda_i f_i(x^+) - m - m\log\mu \\
&\overset{(e)}{=} \mu t f_0(x) - \mu t \left(f_0(x^+) + \sum_{i=1}^{m} \lambda_i f_i(x^+) + \nu^T\left(Ax^+ - b\right)\right) - m - m\log\mu \\
&\overset{(f)}{\leq} \mu t f_0(x) - \mu t g\left(\lambda,\nu\right) - m - m\log\mu \\
&\overset{(g)}{=} m(\mu - 1 - \log\mu)
\end{aligned}
$$

where (a) follows from the expression for the logarithmic barrier $B(x) \equiv -\sum_{i=1}^{m} \log\left(-f_i(x)\right)$, (b) follows from the equalities $\lambda_i = -1/\left(t f_i\left(x\right)\right)$ shown in [45], (c) follows from logarithm rules , (d) follows from the inequality $\log(a) \leq a - 1$ for $a > 0$, (e) holds since $Ax^+ = b$, (f) follows from the definition of the dual function given in [45]

$$
g\left(\lambda,\nu\right) = \inf_z \left(f_0(z) + \sum_{i=1}^{m} \lambda_i f_i\left(z\right) + \nu^T\left(Az - b\right)\right) \leq f_0(x^+) + \sum_{i=1}^{m} \lambda_i f_i(x^+) + \nu^T\left(Ax^+ - b\right)
$$

, and (g) follows from the equality $g\left(\lambda,\nu\right) = f_0(x) - m/t$. The conclusion is that

$$
N_{\text{inner}} \leq \frac{m\left(\mu - 1 - \log\mu\right)}{\gamma} + c. \tag{2.42}
$$

Combining (2.40) and (2.42) results in the following bound on the total number of iterations :

$$
\begin{aligned}
N_{\text{total}} &= N_{\text{outer}} N_{\text{inner}} + N_{\text{initial}} \\
&\leq \left\lceil \frac{\log\left(m/\varepsilon t^{(0)}\right)}{\log\mu} \right\rceil \left(\frac{m\left(\mu - 1 - \log\mu\right)}{\gamma} + c \right) + N_{\text{initial}}
\end{aligned}
$$

2.6 Numerical results

In this section, we provide numerical results concerning the convergence of Newton's method. It is shown that although the improved bounds on the number of Newton iterations are 1-2 orders tighter compared to the original bounds, they are still very loose, especially in the damped phase. Using the empirical results, we point out what steps in the derivation should be improved in order to further tighten the bounds. The numerical results are shown for an un-constrained minimization of an s.c. function $f : \mathbb{R}^n \to \mathbb{R}$ of the form

$$f(x) = c^T x - \sum_{i=1}^{m} \log(b_i - a_i^T x) \tag{2.43}$$

solved using Newton's method with tolerance $\varepsilon = 10^{-10}$. The problem parameters a_i and b_i were randomly generated and the problem's size was $(m, n) = (1200, 450)$.

2.6.1 Predetermined step size t

The optimization problem described above was solved using Newton's method with a pre-determined choice of t, as given in Theorem 4 with parameter $\eta = 0.35$ and as given in Theorem 5 with parameter $\eta = 0.8$.

As seen in Figure 2.11, the two methods show similar behavior (with minor differences at the quadratic phase). The reason is that Newton's method iterates most of the time in the damped phase. In this phase the choice of t is exactly the same for both choices of pre-determined t. We conclude that the improved choice of t is mainly useful for tightening the bound on the number of iterations. Since the two choices iterate almost the same, in the following we shall only analyze the numerical behavior of Theorem 4 with parameter $\eta = 0.35$. Figure 2.12 plots the convergence of a randomly generated problem (A single trial is shown that represents the general behavior of the randomly generated problems). The problem was solved after 46 Newton iterations, of which the first 42 iterations are included in the damped phase (i.e. $\lambda > \eta = 0.35$) and the last 4 iterations are included in the quadratic phase (i.e. $\lambda < \eta = 0.35$). As expected, the damped phase is characterized with slow convergence, while the convergence in the quadratic phase is very rapid.

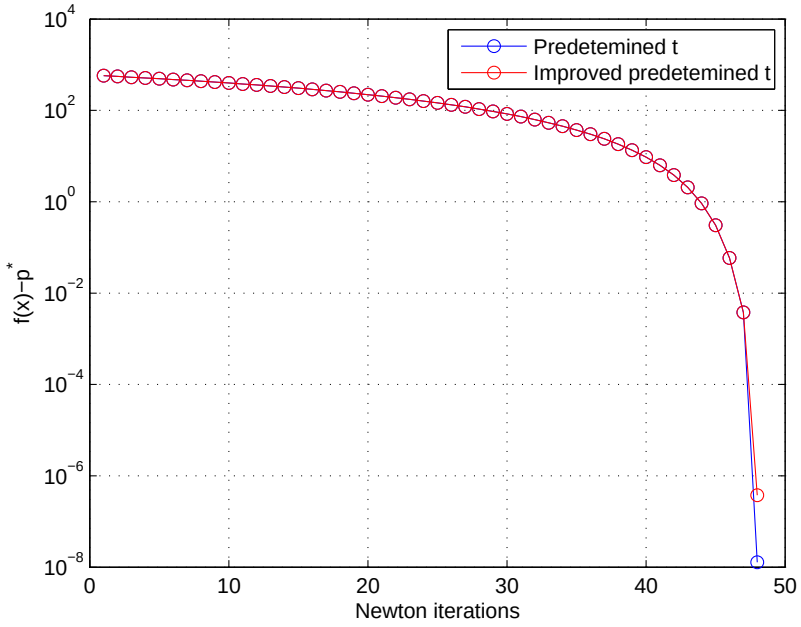

Figure 2.11: Convergence of Newton's method with pre-determined t as given in Theorem 4 with parameter $\eta = 0.35$. The results are given for an un-constrained minimization, as given in (2.43) and as given in Theorem 5 with parameter $\eta = 0.8$. The two choices of t show minor differences

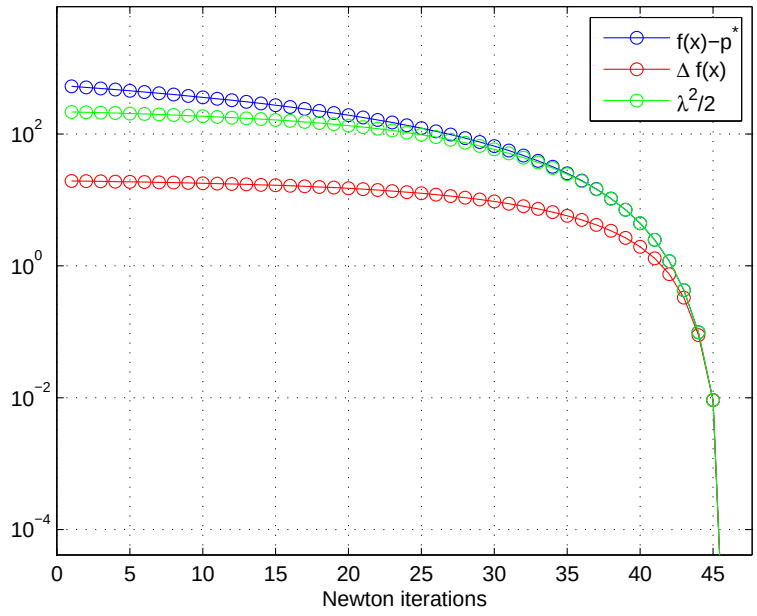

Figure 2.12: Convergence of Newton's method with pre-determined t as given in Theorem 4 with parameter $\eta = 0.35$. The results are given for an un-constrained minimization, as given in (2.43). The figure plots the distance from the infimum $f(x) - p^*$, the decrease in f at each iteration Δf and half the square of the Newton decrement $\lambda^2/2$.

Applying the bound in (2.21) for the simulated problem results in the following bound :

$$N_{\text{total}} = N_{\text{Damped}} + N_{\text{quad}} \leq 20(530) + 7 = 10607.$$

The bound for N_{Quad} is relatively tight (7 iterations as opposed to 4), however the bound for N_{Damped} is still very loose.

From the numerical results of $f(x) - p^*$ compared to its lower bound $\lambda^2/2$ used in the derivation of the Theorem. It turns out that this bound is relatively tight. The bound tightens as iterations increase and we enter the quadratic phase. This bound was used in the derivation of the bound for the quadratic phase. The tightness in this phase agrees with the overall tightness of the bound on the number of Newton iterations in the quadratic phase.

In the following, the reasons for the loose bound for the damped phase are investigated. In the proof of Theorem 4, the number of Newton iterations in the damped phase was bounded by division of $f(x) - p^*$ by a bound for the convergence rate at each iteration $\Delta f(x^{(n)}) > \lambda(x^{(n)}) - \log(1 + \lambda(x^{(n)}))$ (where $\lambda(x^{(n)})$ was globally bounded by $\lambda(x^{(n)}) > \eta$). Figure 2.13 plots this bound normalized by the numerical result for $\Delta f(x^{(n)})$. It is evident that the bound is extremely loose, which explains why the bound on the number of iterations is loose. Also shown in the figure is the same bound only this time $\lambda(x^{(n)})$ was not globally bounded with $\lambda(x^{(n)}) > \eta$ but extracted from the numerical results. With $\lambda(x^{(n)})$ not globally bounded, the bound on $\Delta f(x^{(n)})$ is relatively tight.

We conclude that the bound on the number of Newton iterations during the damped phase is loose, mainly because of the usage of a global bound on $\lambda(x^{(n)}) < \eta$. We can assume that if a tight bound on $\lambda(x^{(n)})$ at each iteration was used, then the bound on the number of iteration would be only 10 to 20 percent larger than the numerical results.

Also shown in Figure 2.13 is the Taylor approximation for the rate of convergence $\Delta\tilde{f}(t) = f(x) - f(x + t\Delta x_{\text{nt}}) \cong t(1 - t/2)\lambda^2$. For $t = \frac{1}{1+\lambda}$ this approximation proved to be extremely accurate, with error $< 0.05\%$. The important thing to notice is that it has the same general behavior as our bounds for $\Delta\tilde{f}(t)$. This shows that the worse-case behavior is similar to the average case behavior, thus our bounds predict the general behavior of the convergence rate. This approximation can be used in an average case analysis of the convergence rate of the Newton method.

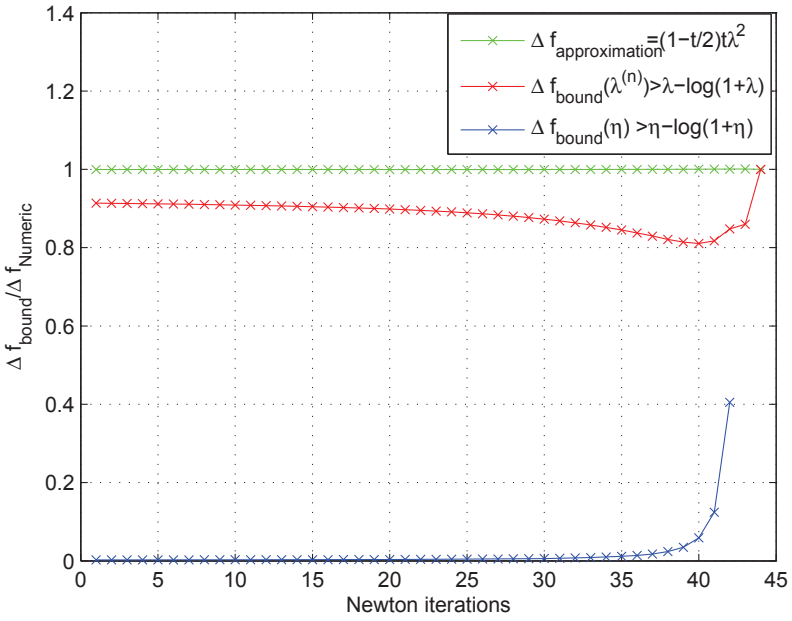

Figure 2.13: Bounds for $\Delta f(x)$ normalized by the Numerically simulated $\Delta f(x)$. The results are for a function of the form given in (2.43) with $t = \frac{1}{1+\lambda}$

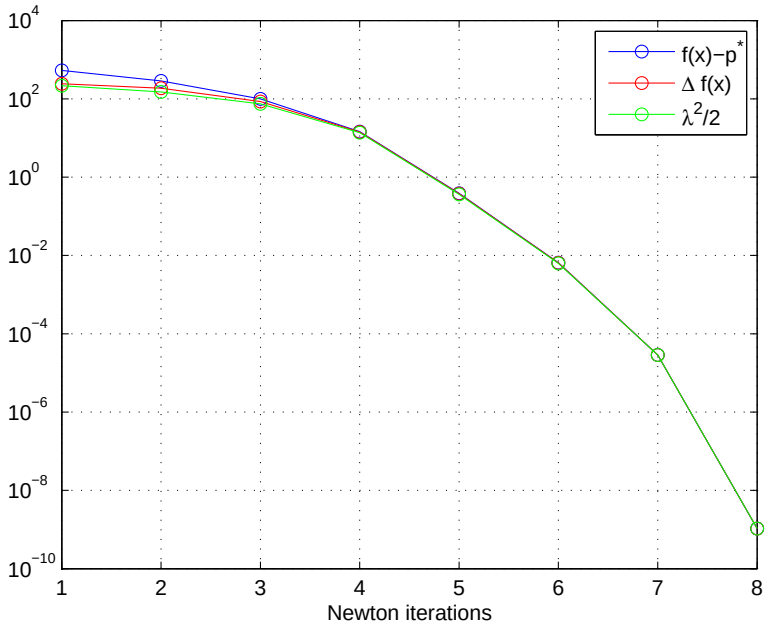

Figure 2.14: Convergence of Newton's method with backtracking line-search. The figure plots the distance from the infimum $f(x) - p^*$, the decrease in f at each iteration Δf and half the square of the Newton decrement $\lambda^2/2$.

2.6.2 Backtracking line-search

We now examine the bound on the number of Newton iterations in Theorem 6 when using backtracking line-search with line-search parameters $\alpha = 0.2$ and $\beta = 0.8$. Figure 2.14 plots the convergence of a randomly generated problem (for comparison the same problem simulated in the subsection about predetermined t is simulated here). The problem is solved after 8 Newton iterations, of which the first 4 iterations are included in the damped phase (i.e. $\lambda > \eta = 0.35$) and the last 4 iterations are included in the quadratic phase (i.e. $\lambda < \eta = 0.35$). Applying the bound in (2.25) for the simulated problem results in $N_{\text{total}} = N_{\text{Damped}} + N_{\text{quad}} \leq 37 * (530) + 11 = 19621$. The bound for N_{Quad} is relatively tight (11 iterations as opposed to 4), however the bound for N_{Damped} is still very loose. In the following, the reasons for the loose bound for the damped phase are investigated. In order to bound the number of iterations in the damped phase, we used the backtracking exit condition as a bound on the convergence rate at each iteration $\Delta f(x) > t\alpha\lambda^2$. We note that in the proof, the

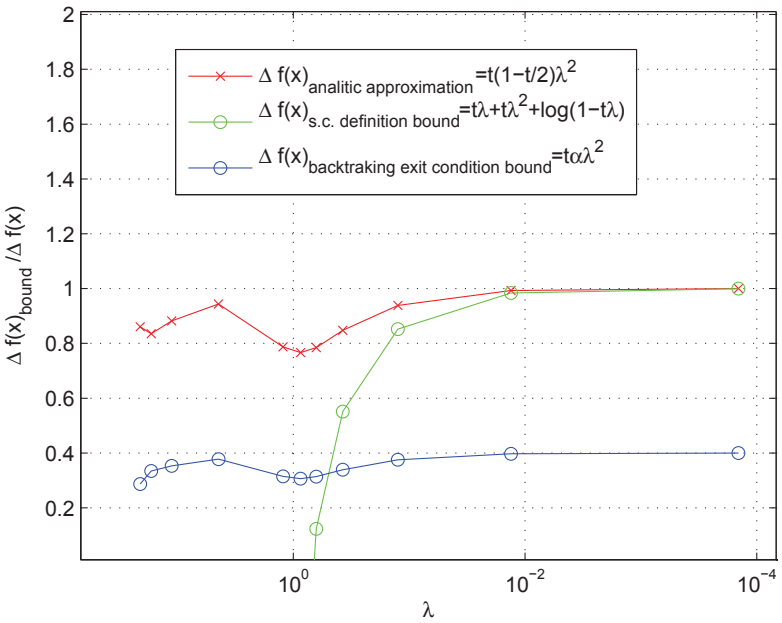

Figure 2.15: Lower bounds for $\Delta f(x)$ normalized by the simulated $\Delta f(x)$. The function is of the form given in (2.43) solved with backtracking line-search

values of λ and t were globally bounded.

Figure 2.15 plots several bounds on $\Delta f(x)$, normalized by the numerical result for $\Delta f(x)$. Similarly to the case of predetermined t, the use of global bounds for λ and t is the main reason for the bound being loose. In order to investigate other reasons, the bounds plotted in the figure are given with λ and t extracted from the numerical results. The figure suggests that the backtracking exit condition bound has the same general behavior as the numeric $\Delta f(x)$, but it is approximately $\alpha/0.5$ times looser. We conclude that in practice as shown later, the exit condition is usually satisfied even without multiplying t by β which results in $t \approx 1$. Another potential reason why the bound might be loose is the case that the minimized function is not minimal s.c.. The bound on $\Delta f(x)$ which follows from the s.c. definition is also plotted. It can be seen that for small λ the s.c. definition bound predicts $\Delta f(x)$ with great accuracy. This means that at the vicinity of x^*, the function $f(x)$ is a minimal s.c. function. Thus the bound can not be improved by simply multiplying the function with a normalization constant as described in remark 5.

Since the backtracking exit condition bound $\Delta f(x) > t\alpha\lambda^2$ is proportional to t, it is important that the bound on t is tight. Figure 2.16 plots the numerical result for t and the bound on t_{bk} using (2.36) and $t_{\text{bk}} = \beta^{\lceil \log_\beta(t_{\text{exit}}) \rceil}$. As mentioned before, numerically, the backtracking line-search algorithm usually terminates at the first iteration resulting in $t \approx 1$. However, the lower bound is much lower, predicting multiple multiplication by β. We note that this bound was derived almost directly from the s.c. definition, thus it is surprising it is not tight. The reason lies in the domain of the s.c. inequality in terms of $\Delta f(x) \geq t\lambda + t\lambda^2 + \log(1 - t\lambda)$ which is $t\lambda < 1$. As $t \to 1/\lambda$ the right hand side of the inequality tends to minus infinity making the inequality useless. This tendency to infinity causes the bound to choose values of t which comply to the artificial constraint of $t < 1/\lambda$, whereas in practice larger values of t also satisfy the backtracking condition and therefore are favorable. This means that any bound derived from the s.c. inequality cannot by tight for $\lambda > 1$. Since this is the case during the damped phase, the bounds get loosen. If in the future better bounds on $\lambda^{(n)}$ at each iteration are found then the fact that the bound for t is not tight for $\lambda > 1$ might be problematic because the overall bound will still be loose. In this case, only a combined improvement (i.e., a tight bound on $\lambda^{(n)}$ at each iteration and a bound on t that does no suffer from the artificial constraint of $t < 1/\lambda$) will yield a bound on the number of iterations in the damped phase that is tight.

Number of Newton iteration vs. α and β

In the following, we explore the behavior of the improved bound on the number of Newton iterations when changing the parameters α and β.

- Dependence on α : Numerical results show little, if any dependence of α on the number of Newton iterations. On the other hand the bound suggests that the number of iterations has a minimum around $\alpha = 0.2$ (See Figure 2.7). This can be explained by the empiric fact that $\Delta f(x) \approx 0.5\lambda^2$ even for $t = 1$. This means that for most cases the backtracking exit condition is satisfied regardless the value of α. The bound shows stronger dependency upon α because it assumes multiplications by β are required in order to satisfy the backtracking exit condition.

- Dependence on β : The analytic bound suggests that the number of Newton

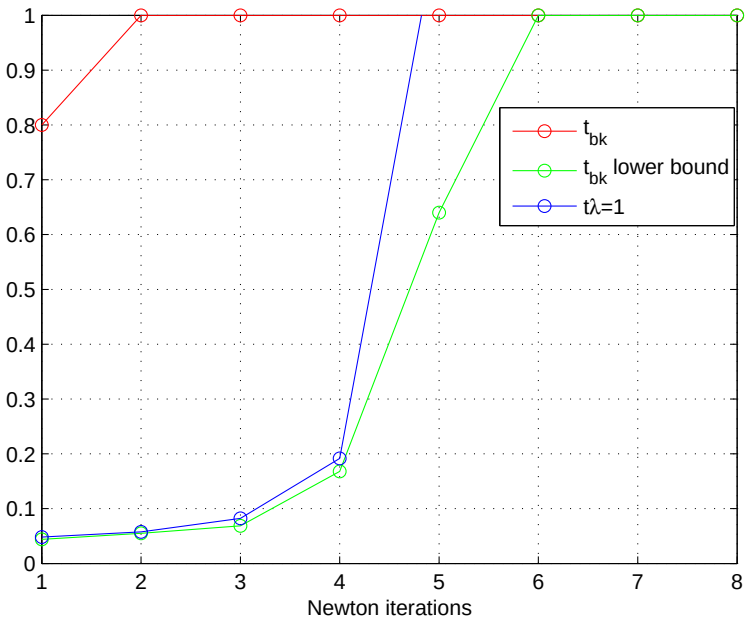

Figure 2.16: Numerical results for t derived using backtracking line-search for a function of the form given in (2.43). The plot includes the numerical value of t_{bk}, the bound on t_{bk} using (2.36) and $t_{\text{bk}} = \beta^{\lceil \log_\beta(t_{\text{exit}}) \rceil}$. For reference, the function $t = 1/\lambda$ is also plotted

iterations scales like $\frac{1}{\beta}$. Therefore the number of iteration decrease as $\beta \to 1$. As seen in Figure 2.17, in practice the behavior is more complex and has the following properties

- Weak dependence on β for most values of β. This can be explained by the fact that usually $t = 1$ is chosen and no multiplications by β are required. In those cases, the value of β is meaningless.
- $\beta \to 0$: The number of iterations increases rapidly (as the analytic bounds suggests). This is because if a multiplications by β occurs, the value of t is reduced dramatically. Therefore a very small Newton step is taken. This in turn slows down the convergence.
- $\beta \to 1$: The number of iterations increases slightly. This behavior is not intuitive, since $\beta \to 1$ will follow in a very fine reduction in the value of t until the backtracking condition is satisfied. Therefore, the largest possible step is taken and a fast convergence is expected. However, t might be multiplied by β in order to get inside the domain of $f(x)$. In this case $\beta \to 1$ will result in a step size that will lead us to the boundary of the domain. Since near the boundary of the domain, $f(x)$ is ill conditioned we can expect slow convergence.

2.6.3 Comparison of backtracking line-search to predetermined step size t

Comparing the bound on the number of Newton iterations with predetermined t to the bound with backtracking line-search, one could think that the solver using predetermined t will converge 10 times faster. Numerical results given in Figure 2.18 show the exact opposite. The backtracking line-search proves to be more efficient numerically. We conclude that optimizing t in order to minimize the bound on the number of Newton iterations (as done in the case of predetermined t) does not necessarily means that this is an optimal choice. The optimization was artificial because the bound on the number of iterations does not reflect the full behavior of the number of Newton iteration. The reason for this is that the bounds used, does not allow $t > 1/\lambda$, where in practice $t\lambda >> 1$ gives better results in the damped phase. In practice the

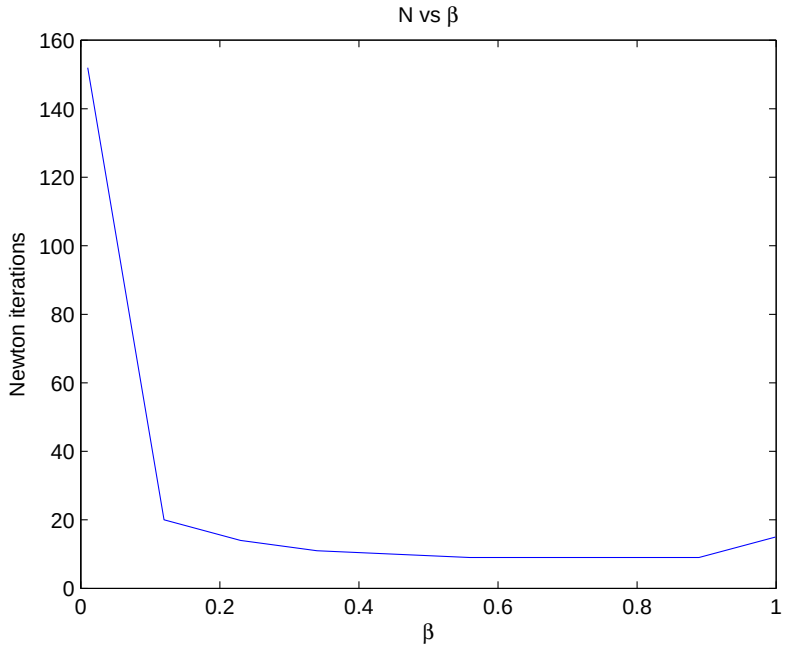

Figure 2.17: Number of Newton iterations (N) versus the value of β in the backtracking line-search

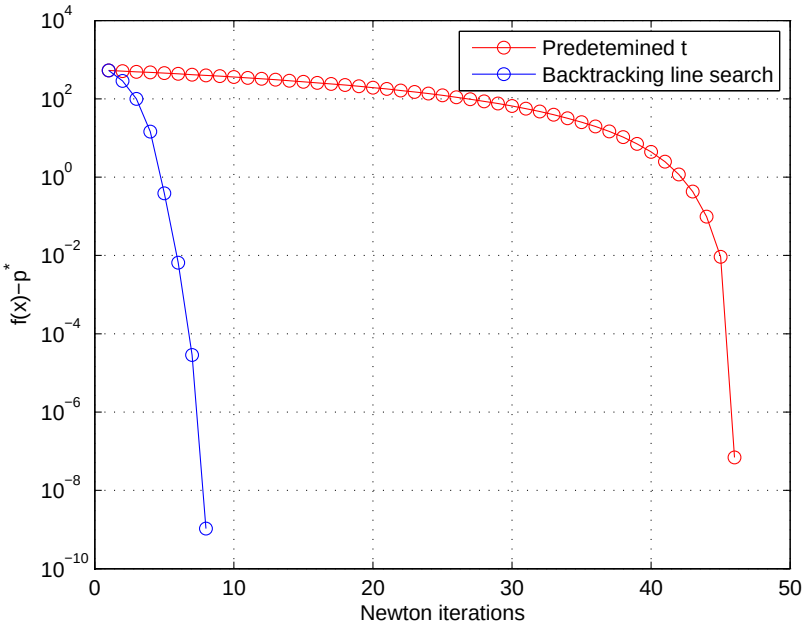

Figure 2.18: Convergence rate of $f(x) - p^*$ for a problem solved using Newton's method. It is evident that the usage of the backtracking line-search results in less Newton iterations compared to the usage of predetermined step size t.

backtracking algorithm does not suffer from this artificial constraint, thus it converge faster during the damped phase. s

2.6.4 Conclusions regarding the numerical analysis

In the following, we summarize some of the main results and conclusions derived throughout the numerical analysis.

- Although tighter compared to previous bounds, the numerical results show that the improved bounds are still very conservative. Using a numerical analysis, we point out the two main reasons that keep the bound from being tight

    1. Global bound for λ : The improved bounds use $\lambda > \eta$ in order to bound λ for all the iterations during the damped phase. The lack of a tight bound for $\lambda^{(n)}$ at each iteration is the main reason why the bounds are not tight at the damped phase.

2. Domain of bounds : The bounds used have a domain of $t\lambda < 1$ whereas in practice it is preferable to choose $t\lambda > 1$ in the damped phase. This artificial constraint on t causes the choice of the predetermined t to be relatively far from optimal. As a result the backtracking bound for t_{bk} is very loose for large values of λ, thus causing the bound on the convergence rate $\Delta f(x) > t\alpha\lambda^2$ to be loose.

- The predetermined choice of t results in a tighter bound as compared to the backtracking bound, however numerically, the algorithm is less efficient. We conclude that the predetermined choice optimizes the bound on the number of Newton iterations but not necessarily the practical efficiency of the solver. Thus, the optimized value of t is an artifact of the bounding method used. It is last as an open question whether this bound is also valid for backtracking line-search.

- It was shown empirically that for functions of the form (2.43) the Taylor series approximation $\Delta f(t) \cong t(1 - t/2)\lambda^2$ predicts the convergence rate with great accuracy. This expression has similar behavior compared to our bounds for $\Delta\tilde{f}(t)$. This shows that the worse-case behavior is similar to the average case behavior, thus our bounds predict the general behavior of the convergence rate. This approximation can be used in an average case analysis of the convergence rate of the Newton method.

Chapter 3

Complexity analysis of IPM-based LP decoders for binary linear block codes

3.1 Short overview

In recent years, the idea of using Linear Programming (LP) as a decoder for linear codes has attracted the attention of many researchers. This alternative decoding algorithm is based on the fact that Maximum-Likelihood Decoding (MLD) can be seen as a combinatorial optimization problem. In the following, we give some background on the field of decoding binary linear block codes using Linear Programming. The decoder presented is based on interior-point methods described in the previous chapter. Using the tightened bounds on the number of Newton iterations, we provide complexity bounds on the IPM-based LP decoder.

3.2 LP decoding background

In this section, we review the basics of LP decoding. The background provides basic notations and major theorems that are essential for the analysis in this work.

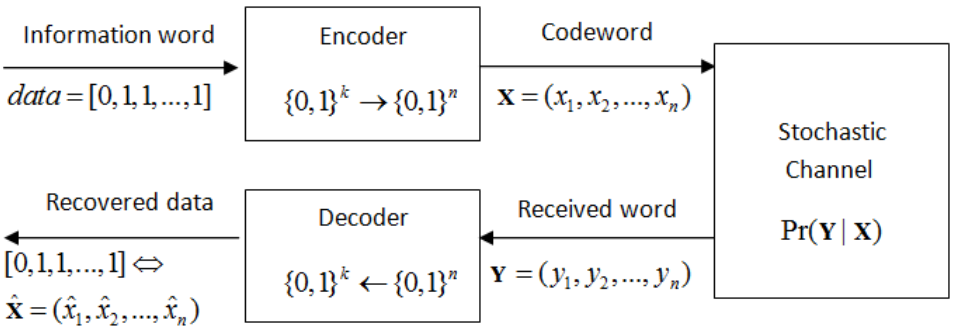

Figure 3.1: A high-level model of a communication link. The information word is encoded into a codeword that contains redundant information. The codeword is sent through the channel (transmission medium) where it is being corrupted. The decoder applies an algorithm that tries to recover the original information from the received word.

3.2.1 Linear block codes

In this thesis, we shall focus on binary linear block codes. A binary code $\mathcal{C}$ of length n is a linear block code if $\mathcal{C}$ is a linear subspace of the vector space $\{0,1\}^n$. Recall that a linear subspace of a vector space is any subspace closed under addition and scalar multiplication, so a binary code $\mathcal{C}$ is linear if the following conditions hold

- All-zero codeword : $0^n \in C$, i.e., the all-zero word belongs to the code.
- Linear property : For all pairs of codewords, if $\mathbf{x}', \mathbf{x} \in \mathcal{C}$, then $(\mathbf{x}' + \mathbf{x})_{mod2} \in \mathcal{C}$ where the mod 2 addition is done bit-wise.

Therefore, the code is a linear subspace of a vector space. This subspace can be specified by the basis of the subspace. For our purposes, the basis is a linearly independent set of codewords $\mathcal{B} = \mathbf{x}^{(1)}, ..., \mathbf{x}^{(k)}$, each of length n, such that every codeword in $\mathcal{C}$ can be expressed as the sum of a subset of codewords in that set. The **generator matrix** G of a binary linear code is an $n \times k$ binary matrix whose columns are the basis vectors. The encoder for a binary code simply multiplies the

information word by the generator matrix G. Therefore,

$$\mathcal{C} = \left\{ G\tilde{x}^T : \tilde{x} \in \{0,1\}^k \right\}.$$

Every linear subspace $\mathcal{C}$ of dimension k has an orthogonal linear subspace $\mathcal{C}^\perp$ of dimension $n-k$. This new subspace $\mathcal{C}^\perp$ can be though of as a new code, and is often called the dual code to $\mathcal{C}$. This dual code also has a basis, and so we can write down a generator matrix H for this code as well. Since we have $< \mathbf{x}, \mathbf{x}^\perp >= 0$ for all $\mathbf{x} \in \mathcal{C}$ and $\mathbf{x}^\perp \in \mathcal{C}^\perp$, it must be the case that for all $\mathbf{x} \in \mathcal{C}$, we have

$$H\mathbf{x} = 0,$$

and so we have

$$HG^T = 0.$$

The matrix H is called **the parity-check matrix** of the code $\mathcal{C}$, and has the property that a word $\mathbf{x}$ is in the code if and only if $\mathbf{x}$ is orthogonal to every row of H. The matrix H is called the parity-check matrix because every row induces a parity-check on the codeword.

LDPC codes

Low-density parity-check (LDPC) codes are a family of linear codes with a sparse parity-check matrix. Mathematically, a family of codes (parameterized by their block length n) is considered to be a family of LDPC codes if the maximal Hamming weight of the rows and columns of the parity-check matrix stay constant as n grows.

We say a code is a regular LDPC$(n, d_\mathrm{v}, d_\mathrm{c})$, if the rows have a constant Hamming weight denoted by d_c and the columns have a constant Hamming weight denoted by d_v. The parameters d_c, d_v are also called the check-node and variable-node degree respectfully. A more general family of LDPC codes is the $(n, \lambda(x) \triangleq \sum_i \lambda_i x^{i-1}, \rho(x) \triangleq \sum_i \rho_i x^{i-1})$ irregular LDPC family, where the rows' and columns' weights are not constant. Considering the factor graph $\mathcal{G}$ of the code, λ_i and ρ_i denote the fraction of edges attached, respectively, to variable and parity-check nodes of degree i.

3.2.2 The ML decoder

The design of a decoder is perhaps the most difficult task in the design of an error-correcting code, especially those that approach the theoretical limits. Given a particular code $\mathcal{C}$, a natural question to ask is: what is the best possible decoder, if our goal is to minimize the word error probability (WER)? The maximum-a-posteriori (MAP) codeword $\mathbf{x}_{\text{MAP}}$ is the one that was most probably transmitted, given the received vector $\mathbf{y}$:

$$\mathbf{x}_{\text{MAP}} = \arg\max_{\mathbf{x}\in\mathcal{C}} \Pr\left[\mathbf{x} \text{ transmitted}|\mathbf{y} \text{ received}\right]$$

Using Bayes' rule, and the assumption that all information words have equal probability, the MAP codeword is the same codeword $\mathbf{x}$ that maximizes the probability $\mathbf{y}$ was received, given that $\mathbf{x}$ was sent:

$$\mathbf{x}_{\text{ML}} = \arg\max_{\mathbf{x}\in\mathcal{C}} \Pr\left[\mathbf{y} \text{ received}|\mathbf{x} \text{ transmitted}\right]$$

An ML decoder is a decoder that always finds the ML codeword. This is also often called optimal decoding. For most codes, there is no known polynomial-time ML decoding algorithm. In fact, the problem is in general NP-hard, and it remains NP-hard for many families of codes used in practice. Therefore in most cases, one would look for a sub-optimal (but efficient) decoder. The goal then is to show that the WER of that decoder is still low. For example, the decoders most often used for LDPC codes are based on *belief-propagation* algorithms (e.g., sum-product algorithm) [43], where messages are iteratively sent across the edges of the factor graph of the code. This decoder finds the ML codeword provided that the Tanner graph of the code contains no cycles. However, cycle-free block codes are known to have bad performance [47], thus when using this algorithm for LDPC codes with cycles, we search for a sub-optimal (but efficient) decoder.

The ML decoder as a min-sum optimization problem

Given a particular received word $\mathbf{y}$, we define the log-likelihood ratio ℓ_i of a code bit x_i to be:

$$\ell_i(y_i) = \ln\left(\frac{\Pr[y_i|x_i = 0]}{\Pr[y_i|x_i = 1]}\right).$$

The sign of the log-likelihood ratio determines whether the transmitted bit x_i is more likely to be a 0 or a 1. If x_i is more likely to be a 1 then ℓ_i will be negative. If x_i is more likely to be a 0 then ℓ_i will be positive.

Theorem 8 [ML decoder as a min-sum optimization problem] For any binary-input memoryless channel, the codeword of minimum cost is the Maximum-Likelihood codeword.

$$\mathbf{x}_{\mathrm{ML}} = \arg\max_{\mathbf{x}\in\mathcal{C}} \Pr\left[\mathbf{y}|\mathbf{x}\right] = \arg\min_{\mathbf{x}\in\mathcal{C}} \left(\sum_{i=1}^{n} \ell_i x_i\right)$$

Proof: See Section 2.5 in [21].

This shows that the ML decoder can be written as an optimization problem. However, since the feasible set is integer, in general no efficient search methods exist and the decoder needs to calculate the cost function for all 2^k possible codewords.

3.2.3 Relaxed ML decoder

As stated before, the ML decoder can be written as an optimization problem. However, in general its complexity is practically infeasible. In order to reduce the complexity of decoding, a technique called relaxation is applied. The basic concept of relaxation is to relax the definition of the feasible set from a discrete set to a subset of an n-dimensional real space $\mathbb{R}^n$. For example, for a combinatorial optimization problem written in the form of integer linear programming, elimination of the constraint that "a feasible point is integral" leads to a linear programming (LP) problem that can be efficiently solved by a simplex method or an interior point method.

In our relaxation, we first let the set of feasible values to be between zero and one, instead of binary. We will define some additional linear constraints on the variables that are a function of the structure of the code itself, and obtain a feasible set which is a polytope $\mathcal{P} \subseteq [0,1]^n$. It is shown in [21], that the minimum of an LP problem with a feasible set is a vertex of $\mathcal{P}$. An extreme would be to choose $\mathcal{P}$ such that $\mathcal{V}(\mathcal{P}) = \mathcal{C}$. In this case, the relaxed LP decoder will output the codeword with minimum cost codeword which is the ML codeword. Due optimal, this polytope is too complex to represent for any code for which ML decoding is NP-hard. In order to reduce the

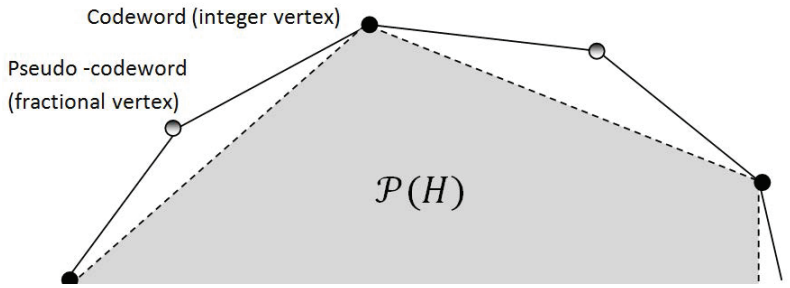

Figure 3.2: A graphical representation of a proper polytope. The relaxation of the domain creates fractional vertices called pseudo-codewords. Since the LP decoder does not differ between codewords and pseudo-codewords, this fractional vertices are the main reason for decoding errors.

computational complexity, we can choose a structured polytope that approximate the optimal polytope. The choice of the polytope is now a tradeoff between performance and complexity. An indicator for a good polytope is the property of properness.

Definition 10 [Proper polytope] We call a polytope $\mathcal{P}$ proper, if the set of integral points in $\mathcal{P}$ is exactly the set $\mathcal{C}$ of codewords.; i.e.,

$$\mathcal{P} \cap \{0,1\}^n = \mathcal{C}.$$

Since the integer vertices of a proper cone are the codewords, then if an LP decoder outputs an integer solution (codeword) then we know it outputs the ML codeword. This property is called the ML certificate property. It is one of the unique advantages of LP decoding.

As illustrated in Figure 3.2 , not all vertices are integers, therefore the decoder might output a fractional solution. This fractional solutions are called pseudo-codewords and are obviously not codewords. Since the LP decoder does not differ between codewords and pseudo-codewords, this fractional vertices are the main reason for decoding errors.

A popular choice for a proper cone for LDPC codes is a relaxed polytope of the convex hull of $\mathcal{C}$. We refer to the following choice as the fundamental polytope [21],[51].

Definition 11 [The fundamental polytope] Consider a code with parity-check matrix H, and let

$$A_i = \{j \in [1,n] : h_{ij} = 1\}$$

for $i \in [1, m]$, where h_{ij} is the (i, j)-element of H. The set $T_i (i \in [1, m])$ is the set of all subsets of odd size in A_i, namely

$$T_i = \{S \subset A_i : |S| \text{ is odd}\}.$$

The constraints for $\mathbf{x} = (x_1, x_2, ..., x_n) \in \mathbb{R}^n$,

$$\forall i \in [1, m], \quad \forall S \in T_i, \quad 1 + \sum_{t \in S} (x_t - 1) - \sum_{t \in A_i \setminus S} x_t \leq 0 \tag{3.1}$$

and

$$\forall j \in [1, n], \quad 0 \leq x_j \leq 1 \tag{3.2}$$

are referred to as the parity constraints and the box constraints, respectively. The fundamental polytope $\mathcal{P}(H)$ is the polytope defined by

$$\mathcal{P}(H) = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{x} \text{ satisfies constraints (3.2),(3.1)}\}$$

For each one of the m parity-check equations that a codeword must hold a set of $2^{d_c - 1}$ inequalities constraints are set. This parity constraints basically forbid words with odd Hamming weight at the indexes where $h_i j = 1$. The fundamental polytope can be shown to be a proper polytope, therefore the ML certificate property holds.

3.3 Application of interior-point methods to LP decoding of binary linear block codes

Here, we apply the interior-point method based on log-barrier functions to the relaxed MLD problem. We provide an IPM-based LP decoder similar to the decoder given in [50],[51]. Based on the complexity bounds derived in previous sections, we derive an upper bound on the number of Newton iterations for the LP decoder. Similar analysis with loosened bounds was presented in [50]. This analysis was valid only for regular LDPC codes using the backtracking line-search whereas this analysis uses tightened bounds, and it applies to general non-regular codes with backtracking, exact or predetermined line-search. Next, we discuss the properties of the derived bound in the context of LDPC codes.

3.3.1 IPM-based LP decoder

Consider a received word $\mathbf{y}$, denote $\ell_i(y_i) = \ln\left(\frac{\Pr[y_i|x_i=0]}{\Pr[y_i|x_i=1]}\right)$ as the log-likelihood ratio (LLR) with respect to codeword bit x_i. Using Theorem 8, the LP decoder objective function is given by

$$f(\mathbf{x}) = \sum_{i=1}^{n} \ell_i(y_i) x_i. \tag{3.3}$$

If we use the fundamental polytope as in definition 11, the resulting log-barrier function $B(\mathbf{x})$ includes parity-check constraints and box constraints and is given by

$$\begin{aligned} B(\mathbf{x}) = \quad & \sum_{i\in[1,m]} \sum_{S\subset T_i} \ln\left[-\left(1+\sum_{t\in S} x_t - 1 - \sum_{t\in A_i\setminus S} x_t\right)\right] \\ & - \sum_{j\in[1,n]} \ln\left[x_j\right] - \sum_{j\in[1,n]} \ln\left[-(x_j-1)\right]. \end{aligned} \tag{3.4}$$

Therefore, in order to perform decoding of the received value, the IPM-based decoder computes a sequence of points on the central path $\mathbf{x}^*(t)$. Each point is computed by applying Newton's method to the following un-constrained problem

$$\mathbf{x}^*(t) = \arg\min\{tf(\mathbf{x}) + B(\mathbf{x})\}.$$

Denote M as the number of inequality constraints. The stopping criterion is satisfied when $\frac{M}{t} < \epsilon$. The box constraints contribute n inequalities, while each parity-check equation contributes 2^{d_c-1} inequalities. Therefore, for a general linear code with distribution $\rho(x)$, the total number of inequalities is

$$M = 2n + \left(\frac{m}{\int_0^1 \rho(x)dx}\right) \sum_{i=1}^{d_c^{\max}} \frac{\rho_i 2^{i-1}}{i} = 2n + m d_c^{\text{avg}} \sum_{i=1}^{d_c^{\max}} \frac{\rho_i 2^{i-1}}{i}. \tag{3.5}$$

As a starting point for the initial centering, a word close to the final solution will help the convergence of the optimization algorithm. An example for such a word is the received word $\mathbf{y}$. However, this word is not necessarily a feasible starting point. Therefore, we use the choice $\mathbf{x}^{(0)} = (1/2, 1/2, ..., 1/2)$ as starting point. The

following lemma justify this choice as a feasible starting point. It is a straightforward generalization of the lemma from [51] to non-regular codes.

Lemma 4 Consider a binary linear block code with parity-check matrix H. If the row distribution $\rho(x)$ satisfies $\rho_2 = 0$ (i.e., the minimum row degree $d_{\mathrm{c}}^{\min}$ is greater then 2), then the point $\mathbf{x}^{(0)} = (1/2, 1/2, ..., 1/2)$ is a feasible initial search point (i.e., $\mathbf{x} \in \mathcal{P}(H)$).

Proof: Clearly, $\mathbf{x}$ satisfies the box constraints (3.2). Therefore, we only need to prove the lemma for the parity constraints (3.1). For any $i \in [1, m], S \in T_i$ we have

$$1 + \sum_{t \in S} (\frac{1}{2} - 1) - \sum_{t \in A_i \backslash S} \frac{1}{2} = 1 - \frac{|A_i|}{2} \leq 1 - \frac{d_{\mathrm{c}}^{\min}}{2} \leq 1 - \frac{3}{2} < 0.$$

The last inequality follows from the assumption that $\rho_2 = 0$, since it implies that $|A_i| \geq 3$.

We summarize the IPM-based LP decoding algorithm as following

Definition 12 [IPM-based LP decoding algorithm] Consider a binary linear block code with parity-check matrix $H \in \mathbb{R}^{m \times n}$ with check node distribution $\rho(x)$. Given a received word $\mathbf{y}$, compute the objective function $f(\mathbf{x})$ using (3.3) and the barrier function $B(\mathbf{x})$ using (3.4). Let $t := t^{(0)}, \mu > 1, \epsilon > 0$, $M = 2n + m d_c^{\mathrm{avg}} \sum_{i=1}^{d_{\mathrm{c}}^{\max}} \frac{\rho_i 2^{i-1}}{i}$ and $\mathbf{x}^{(0)} = (1/2, 1/2, ..., 1/2)$ as the initial starting point.

Repeat the following steps (Outer loop)

1. Centering step : Compute $\mathbf{x}^*(t)$ by minimizing $\Psi_t(\mathbf{x}) = t f(\mathbf{x}) + B(\mathbf{x})$, starting at $\mathbf{x} =: \mathbf{x}^{(0)}$.
   Repeat the following steps (Inner loop)
   (a) *Compute the Newton step* : $\Delta \mathbf{x}_{nt} = -\nabla^2 \Psi_t(\mathbf{x})^{-1} \nabla \Psi_t(\mathbf{x})$.
   (b) *Stopping criterion* : Compute the Newton decrement $\lambda^2 = \Delta \mathbf{x}_{nt}^T \nabla^2 \Psi_t(\mathbf{x}) \Delta \mathbf{x}_{nt}$. If $\lambda^2/2 \leq \epsilon$, set $\mathbf{x}^*(t) = \mathbf{x}$ and quit.
   (c) *Line search* : Choose step-size t_{line} (e.g., backtracking line).
   (d) *Update* : $\mathbf{x} := \mathbf{x} + t_{\mathrm{line}} \Delta \mathbf{x}_{nt}$.

2. Update starting point : $\mathbf{x}^{(0)} := \mathbf{x}^*(t)$.

3. Stopping criterion : Quit if $M/t < \epsilon$

4. Increase t : $t := \mu t$.

If $\mathbf{x}^*(t)$ is an integer then $\mathbf{x}_{\mathrm{LP}} = \mathbf{x}^*(t)$, else report an error.

3.3.2 Complexity analysis of an IPM-based LP decoder

We note that the solved problem is an IECSCM problem, therefore we can apply the complexity bounds we developed earlier in order to derive a bound on the number of Newton iterations.

Theorem 9 [Bound on the number of Newton iteration for an IPM-based LP decoding algorithm] Consider the IPM-based LP decoding algorithm in definition 12. Denote $\ell_{\max}$ as an upper bound on $|\ell_i(y_i)|$. The number of Newton iterations is upper bounded by

$$\begin{aligned} N_{\mathrm{Total}}^{\text{LP decoder}} &= N_{\mathrm{outer}} N_{\mathrm{inner}} + N_{\mathrm{initial}} \\ &\leq \left\lceil \frac{\log\left(M/(\varepsilon t^{(0)})\right)}{\log \mu} \right\rceil \left(\frac{M\left(\mu - 1 - \log \mu\right)}{\gamma} + c \right) + \frac{1/2 t^{(0)} \ell_{\max} n}{\gamma} + c \end{aligned}$$

The numbers γ and c are calculated by excluding the inequality constraints in the original problem and applying the bound in Theorems 4, 5 or 6 depending on the line-search method chosen. The numbers γ and c are extracted by comparing the bound to the form "$N_{\mathrm{Total}} \leq \frac{f(\mathbf{x}^{(0)}) - p^*}{\gamma} + c$".

Proof: The proof follows closely the proof given in [50] for a regular LDPC code using loosened bounds for the number of Newton iterations. We note that the LP decoding algorithm applies to the barrier method of an IECSCM problem. Applying Theorem 7 proves the first part of the bound. Next, we consider the second term of the bound that refers to the number of Newton iterations during the initial centering step, i.e., from $\mathbf{x}^{(0)} = \{1/2, 1/2, ..., 1/2\}$ to $\mathbf{x}^*(t^{(0)})$.

This problem is an UCSCM problem, therefore, depending on the line-search method, Theorem 5, 4 or 6 can be applied.

$$N_{\text{initial}} \leq \frac{\Psi_{t^{(0)}}(\mathbf{x}^{(0)}) - \Psi_{t^{(0)}}(\mathbf{x}^*(t^{(0)}))}{\gamma} + c \tag{3.6}$$

The numerator can be upper bounded by

$$\begin{aligned}
\Psi_{t^{(0)}}(\mathbf{x}^{(0)}) - \Psi_{t^{(0)}}(\mathbf{x}^*) &= t^{(0)} f(\mathbf{x}^{(0)}) + B(\mathbf{x}^{(0)}) - t^{(0)} f(\mathbf{x}^*) - B(\mathbf{x}^*) \\
&\overset{\text{(a)}}{\leq} t^{(0)} f(\mathbf{x}^{(0)}) + B(\mathbf{x}^*) - t^{(0)} f(\mathbf{x}^*) - B(\mathbf{x}^*) \\
&= t^{(0)} \sum_{i=1}^{n} \ell_i (1/2 - x_i^*) \\
&\overset{\text{(b)}}{\leq} \frac{t^{(0)}}{2} \sum_{i=1}^{n} |\ell_i| \\
&\overset{\text{(c)}}{\leq} \frac{t^{(0)}}{2} \ell_{\max} n
\end{aligned}$$

where (a) follows from the fact that $\mathbf{x}^{(0)} = \{1/2, 1/2, ..., 1/2\}$ is the minimizer of $B(\mathbf{x})$ (i.e., $\nabla B(\mathbf{x}^{(0)}) = 0$), (b) follows from $|x_i - 1/2| \leq 1/2$ and (c) follows from the definition of $\ell_{\max}$ as an upper bound for $|\ell_i|$. This result combined with 3.6 proves the required bound on N_{initial}.

Remark 8 For regular LDPC codes with check node degree d_{c}, the expression for the total number of inequalities in (3.5), can be degenerated to

$$M = 2n + m2^{d_{\text{c}}-1} = n(2 + (1-R)2^{d_{\text{c}}-1}).$$

3.3.3 Parameter-optimized complexity bound for LP decoder

In this subsection, we use the bound on the number of Newton iterations derived in Theorem 9 and optimize the search parameters μ and $t^{(0)}$. It turns out we can reduce the complexity of the bound with respect for M if we by make μ and $t^{(0)}$ a function of M.

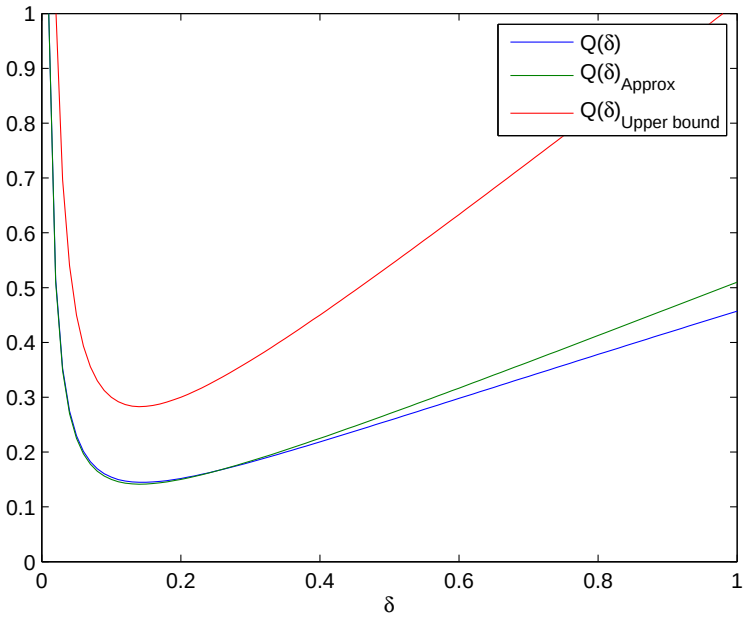

Figure 3.3: The figure plots the function $Q(\delta) \equiv \frac{\delta - \ln(1+\delta) + c\gamma/M}{\ln(1+\delta)}$, its approximation $Q(\delta)_{\text{Approximation}} \equiv \frac{\delta^2/2 + c\gamma/M}{\delta}$ and its upper bound $Q(\delta)_{\text{Upper bound}} = 2Q(\delta)_{\text{Approximation}}$. The plot is given for $c = 6, \gamma = 1/6, M = 100$.

Theorem 10 [Parameter-optimized complexity bound for LP decoder] Consider the bound in Theorem 9, for a code with $M \geq \frac{c\gamma}{20}$. Then, the bound is minimized (approximately) for the search parameters

$$\begin{aligned} \mu^* &= 1 + \sqrt{\frac{2c\gamma}{M}} \\ t^{(0)*} &= \frac{\sqrt{32Mc\gamma}}{n\ell_{\max}}. \end{aligned} \tag{3.7}$$

Using this optimized search parameters, the number of iterations is bounded by

$$N^*_{total} \leq \sqrt{\frac{8cM}{\gamma}} \left[\ln\left(\sqrt{\frac{M}{32c\gamma}} \frac{\ell_{\max} n}{\epsilon} \right) + 1 \right] + c \tag{3.8}$$

Proof: Consider the bound in Theorem 9. It can be verified numerically that for $M \gg c\gamma$, the bound is minimized for values of μ near the value 1.

Therefore, if we set $\mu = 1 + \delta$, then we can simplify the bound with very good accuracy by replacing $\ln(1+\delta)$ with polynomial expressions, i.e., $\frac{\delta - \ln(1+\delta) + c\gamma/M}{\ln(1+\delta)} \cong \frac{\delta^2/2 + c\gamma/M}{\delta}$. Unfortunately, as shown in Figure 3.3 the approximation lower bounds the original expression. By multiplying it by 2, it can be shown that we get an upper bound that holds for $\frac{c\gamma}{M} \leq 20$. This results in a slightly loosen version of the original bound (by factor 2)

$$\begin{aligned} N_{\text{Total}}^{\text{LP decoder}} &\leq \left\lceil \frac{\log\left(M/(\varepsilon t^{(0)})\right)}{\log \mu} \right\rceil \left(\frac{M\left(\mu - 1 - \log \mu\right)}{\gamma} + c \right) + \frac{1/2t^{(0)}\ell_{\max} n}{\gamma} + c \\ &\leq 2 \left(\frac{\log\left(M/(\varepsilon t^{(0)})\right)}{\delta} \right) \left(\frac{M\left(+\frac{\delta^2}{2}\right)}{\gamma} + c \right) + \frac{1/2t^{(0)}\ell_{\max} n}{\gamma} + c. \end{aligned}$$

To get the optimized values presented in 3.7, we nullify the derivative of the loosen bound with respect to $t^{(0)}$ and δ. Using this optimized search parameters, the number of iterations is bounded by (3.8).

Remark 9 Practically, good values of the parameter μ lie in the range 2-100. We would not use the value $\mu^* = 1 + \sqrt{\frac{2c\gamma}{M}}$, which is far too small. Our main interest in this value of μ is that it (approximately) minimizes our (very conservative) upper bound on the number of Newton steps, and yields an overall estimate that grows as $\sqrt{M}$, instead of M. This result is an artifact of the bounding method. Even with the improved values for c, γ, the bound on the inner iterations is very loose and increases very fast with respect to $f(\mathbf{x}) - p^*$. By setting μ very close to 1, the starting point for each inner iteration is very close to the solution on the central path, thus lowering the number of internal iterations required.

3.3.4 Properties of the complexity bound for LP decoder

We are now ready to discuss the behavior of our complexity bound. We will refer to the bound in Theorem 9, and also to its optimized version in Theorem 10. When comparing the two bounds, it is evident that the optimized bound in Theorem 10 increases much slower with respect to the code's parameters.

- [**Number of Inequalities -** M] : The number of inequalities is a measure of the complexity of the fundamental polytope. As the number of inequalities

is increased, the polytope is getting more complex, and better decoding performance can be maintained. The bound in Theorem 9 scales like $O(M \ln M)$, whereas in Theorem 10 the bound scales only like $O(\sqrt{M} \ln M)$. Both bounds show that there exist a trade-off between the decoding performance (represented by the number of inequalities) and the decoding complexity (represented by the number of Newton iterations times the complexity per iteration).

- [**Noise level -** $\ell_{\max}$] : The level of noise in the channel is represented by the value of $\ell_{\max}$. The bound in Theorem 9 scales like $O(\ell_{\max})$, whereas in Theorem 10 the bound scales like $O(\ln \ell_{\max})$.

Next, for the simplicity of the analysis, we shall assume the code is regular with check node degree d_c. In this case the expression for the number of inequality degenerates to

$$M = 2n + m2^{d_c-1} = n(2 + (1 - R)2^{d_c-1}).$$

Next, we substitute this expression in the optimized bound of Theorem 10, and explore its dependence on d_c and n.

- [**Block length -** n] : Error-correcting codes that operate reliably at rates close to the channel capacity suffer from the drawback of having large block-lengths (due to sphere-packing bounds). It is therefore important that the decoding complexity would have a reasonable scaling as a function of the block length n. For general linear codes d_c scales like $O(n)$. Since M scales like $O(2^{d_c})$, then in general the complexity is exponential in n. For LDPC codes d_c is fixed (i.e., it stays fixed irrespectively of n) thus M scales like $O(n)$. Therefore, the bound on the number of iterations scales like $O(\sqrt{n} \ln n)$. However, in each Newton iteration we need to calculate the inverse of the Hessian matrix $\nabla^2 \Psi_t(\mathbf{x}) \in \mathbb{R}^{n \times n}$. For a general code this matrix has no structure, thus the computation requires $O(n^3)$ operations. Fortunately, for LDPC codes, this matrix is sparse, therefore computing its inverse usually requires only $O(n)$ operations [45]. Therefore, the total time complexity scales like $O(n^{1.5} \ln n)$, which is relatively low. We note that this result is higher compared to a result by D. Burshtein [11] that provides an LP decoder with number of iterations that scales like $O(n)$.

- [**Check node degree - $d_{\mathbf{c}}$**] : We note that M scales like $O(2^{d_{\mathrm{c}}})$. This alone makes the number of iterations to be exponential in d_{c}. Therefore, the bound suggests that the proposed LP decoder has long running times for codes with high check-node degree.

Remark 10 We note that there exist alternative polytopes (see [21],[25]) that are more suitable for high-density codes. For these polytopes, the number of inequalities scales like $O(nm + md_{\mathrm{c}}^2 + d_{\mathrm{c}}d_{\mathrm{v}})$. Since M scales like $O(n^3)$, the bound on the number of iterations scales like $O(n^{1.5} \ln n)$. Computing the inverse of the hessian matrix requires $O(n^3)$ operations, thus the bound on complexity scales like $O(n^{4.5} \ln n)$. In a similar manner, since M scales like $O(d_{\mathrm{c}}^2)$ then the bound on the number of iterations scales like $O(d_{\mathrm{c}} \ln d_{\mathrm{c}})$.

Remark 11 We note that the time complexity of the IPM-based LP decoder depends not only on the number of Newton iterations but also on the complexity per iteration. At each Newton iteration, we need to calculate the inverse of the Hessian matrix (i.e., inverting the matrix $\nabla^2 \Psi_t(\mathbf{x}) \in \mathbb{R}^{n \times n}$). In order to bound the overall complexity of the decoder, one needs to bound the complexity involved in that inversion.

3.4 Comparison of the bounds

In the following, we compare the original bound by Wadayama given in [50] to the improved bound given in this thesis. We consider an LDPC(1008,3,6) code transmitted through an AWGN channel with $\frac{E_b}{N_0}$=2.0 dB. Moreover, for the IPM, the following parameters are assumed : $\epsilon_{\text{Inner iterations}} = 10^{-3}, \epsilon_{\text{Outer iterations}} = 10^{-3}, \alpha = 0.3, \beta = 0.5, t^{(0)} = 20$ and $\mu = 20$.

Applying the original bound by Wadayama gives an upper bound of about 10^8 Newton iterations, as compared to about 10^7 Newton iterations when applying the tightened bound for backtracking line-search. In practice, numerical simulations using the backtracking line-search show convergence after about 10^2 Newton iterations. Since in practice, we know that the predetermined step size and the optimized parameters μ^* and $t^{(0)*}$ only increase the number of Newton iterations, then we can use the bound derived for predetermined step size with optimized μ^* and $t^{(0)*}$ as a bound for backtracking line-search. As seen in Table 3.1, applying the bound for the

improved predetermined step size with $t^{(0)} = 20$ and $\mu = 20$ gives a bound of about 10^6 Newton iterations. Applying the bound for the improved predetermined step size with the optimized IPM μ^* and $t^{(0)*}$ gives a much tighter bound of about 10^4 Newton iterations. We conclude that the improved bound for backtracking line-search tighten the bound by factor of 10 and the bound for predetermined step size results in further improvement of factor 10. However, even with these improved bounds, the upper bounds give a very conservative estimate of the number of Newton iterations that does not reflect average case behaviors. In practice, the centering step requires much fewer iterations to achieve reasonable convergence. However, these bounds can be used as a theoretical basis to justify the low complexity of the LP decoder when used to decode LDPC codes. Moreover, although not recommended practically, the optimized parameters μ^* and $t^{(0)*}$ tighten the bound considerably, closing some of the gap between the conservative theoretical bounds and the practical results for the number of Newton iterations.

Source	IPM parameters	Search method	Iterations
Simulations	$(\mu, t^{(0)}) = (20, 20)$	$(\alpha, \beta) = (0.3, 0.5)$	10^2
Original bound [50]	$(\mu, t^{(0)}) = (20, 20)$	$(\alpha, \beta) = (0.3, 0.5)$	10^8
Tightened bound	$(\mu, t^{(0)}) = (20, 20)$	$(\alpha, \beta) = (0.3, 0.5)$	10^7
Tightened bound	$(\mu, t^{(0)}) = (20, 20)$	Pre-determined t	10^6
Tightened bound	Optimized	Pre-determined t	10^4

Table 3.1: Numerical comparison of different bounds on the number of iterations for an IPM-based decoder. We consider an LDPC(1008,3,6) code transmitted through an AWGN channel with SNR 2.0 dB.

Chapter 4

Concentration of measures in LDPC code ensembles

4.1 Short overview

This chapter considers concentration phenomena of LDPC code ensembles. The basic concentration theorem of iterative message-passing decoding asserts that all except an exponentially (in the block length) small fraction of codes perform within an arbitrary small δ from the ensemble average (where δ is a positive number that can be chosen arbitrarily small). Therefore, assuming a sufficiently large block length, the ensemble average forms a good indicator for the performance of individual codes.

In Section 4.2, we provide some mathematical background about Martingales and Azuma's inequality. These mathematical tools will help us to derive concentration results for ensembles of LDPC codes. Next, in Section 4.3, we provide briefly some applications in coding and communication theory of the mathematical tools presented in the previous section. Using Azuma's inequality, we analyze the concentration of measures in LDPC code ensembles.

- Section 4.4 provides a large-deviation analysis of the conditional entropy, and it derives tightened inequalities as compared to the original bound by Méasson et al. [8, Theorem 4].

- Section 4.5 shows concentration results on the number of erroneous variable-to-check messages for Inter-Symbol-interference (ISI) channels. The analysis

provides explicit expression for the exponential rate that is related to the concentration inequalities given in [3, Theorems 1 and 2]. It is shown that particularizing these results for memoryless channels provides tightened concentration inequalities as compared to [29] and [48].

4.2 Mathematical background about Martingales and Azuma's inequality

In this section, we present relevant mathematical background that is essential for the analysis in this work.

4.2.1 Doob's Martingales

This sub-section provides a short background on martingales to set definitions and notation.

Definition 13 [Doob's Martingale] Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. A Doob's martingale sequence is a sequence $X_0, X_1, \ldots$ of random variables (RVs) and corresponding sub σ-algebras $\mathcal{F}_0, \mathcal{F}_1, \ldots$ that satisfy the following conditions:

1. $X_i \in \mathbb{L}^1(\Omega, \mathcal{F}_i, \mathbb{P})$ for every i, i.e., each X_i is defined on the same sample space Ω, it is measurable with respect to the corresponding σ-algebra $\mathcal{F}_i$ (i.e., X_i is $\mathcal{F}_i$-measurable) and $\mathbb{E}[|X_i|] = \int_\Omega |X_i(\omega)| d\mathbb{P}(\omega) < \infty$.
2. $\mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq \ldots$ (where this sequence of σ-algebras is called a filtration).
3. The equality $X_i = \mathbb{E}[X_{i+1}|\mathcal{F}_i]$ holds almost surely (a.s.) for every i.

In this case, it is said that the martingale sequence $\{X_i\}_{i=0}^n$ is adapted to the filtration $\{\mathcal{F}_i\}_{i=0}^n$.

Remark 12 For every i

$$\mathbb{E}[X_{i+1}] = \mathbb{E}\big[\mathbb{E}[X_{i+1}|\mathcal{F}_i]\big] = \mathbb{E}[X_i]$$

so the expectation of a martingale stays constant.

Remark 13 One can generate martingale sequences by the following procedure: Given a RV $X \in \mathbb{L}^1(\Omega, \mathcal{F}, \mathbb{P})$ and an arbitrary filtration of sub σ-algebras $\{\mathcal{F}_i\}$, let

$$X_i = \mathbb{E}[X|\mathcal{F}_i] \quad i = 0, 1, \ldots.$$

Then, the sequence $X_0, X_1, \ldots$ forms a martingale since

1. The RV $X_i = \mathbb{E}[X|\mathcal{F}_i]$ is $\mathcal{F}_i$-measurable, and also $\mathbb{E}[|X_i|] \leq \mathbb{E}[|X|] < \infty$ (since conditioning reduces the expectation of the absolute value).

2. By construction $\{\mathcal{F}_i\}$ is a filtration.

3. For every i

$$\begin{aligned}
&\mathbb{E}[X_{i+1}|\mathcal{F}_i] \\
&= \mathbb{E}\big[\mathbb{E}[X|\mathcal{F}_{i+1}]|\mathcal{F}_i\big] \\
&= \mathbb{E}[X|\mathcal{F}_i] \quad (\text{since } \mathcal{F}_i \subseteq \mathcal{F}_{i+1}) \\
&= X_i \quad \text{a.s.}
\end{aligned}$$

Remark 14 In continuation to Remark 13, one can choose

$$\mathcal{F}_0 = \{\Omega, \emptyset\}, \quad \mathcal{F}_n = \mathcal{F}$$

so that $X_0, X_1, \ldots, X_n$ is a martingale sequence where

$$\begin{aligned}
X_0 &= \mathbb{E}[X|\mathcal{F}_0] = \mathbb{E}[X] \quad (\text{since } X \text{ is independent of } \mathcal{F}_0) \\
X_n &= \mathbb{E}[X|\mathcal{F}_n] = X \ \text{ a.s.} \quad (\text{since } X \text{ is } \mathcal{F}\text{-measurable}).
\end{aligned}$$

In this case, one gets a martingale sequence where the first element is the expected value of X, and the last element of the sequence is X itself (a.s.). This has the following interpretation: At the beginning, we don't know anything about X, so it is initially estimated by its expectation. We then reveal at each step more and more information about X until we can specify it exactly (a.s.).

4.2.2 Azuma's Inequality

Definition 14 [Bounded-difference martingales] Let a sequence of random variables $\{X_i\}_{i=0}^n$ be a martingale. This sequence is said to be a bounded-difference martingale if there exists a fixed vector $d = (d_1, \ldots, d_n)$ of non-negative entries such that the condition

$$|X_i - X_{i-1}| \leq d_i$$

is satisfied a.s. for every $i \in \{1, 2, \ldots, n\}$.

Azuma's inequality forms a useful concentration inequality for bounded-difference martingales [24]. In the following, this inequality is introduced.

Theorem 11 [Azuma's inequality] Let $X_0, \ldots, X_n$ be a bounded-difference martingale with the associated vector d in Definition 14, then

$$\mathbb{P}(|X_n - X_0| \geq r) \leq 2 \exp\left(-\frac{r^2}{2\sum_{i=1}^n d_i^2}\right), \quad \forall\, r > 0.$$

4.3 Some Applications of Azuma's Inequality in Coding Theory

In the following, we shortly provide some examples of the use of Azuma's inequality for coding and communication theory.

4.3.1 Minimum Distance of Binary Linear Block Codes

Consider the ensemble of binary linear block codes of length n and rate R. The average value of the normalized minimum distance is equal to

$$\frac{\mathbb{E}[d_{\min}(\mathcal{C})]}{n} = h_2^{-1}(1 - R)$$

where h_2^{-1} designates the inverse of the binary entropy function to the base 2, and the expectation is with respect to the ensemble where the codes are chosen uniformly at random (see [1]).

Let H designate an $n(1-R) \times n$ parity-check matrix of a linear block code $\mathcal{C}$ from this ensemble. The minimum distance of the code is equal to the minimal number of columns in H that are linearly dependent. Note that the minimum distance is a property of the code, and it does not depend on the choice of the particular parity-check matrix which represents the code.

Let us construct a martingale sequence $X_0, \ldots, X_n$ where X_i (for $i = 0, 1, \ldots, n$) is a RV that denotes the minimal number of linearly dependent columns of a parity-check matrix that is chosen uniformly at random from the ensemble, given that we already revealed its first i columns. Based on Remarks 13 and 14, this sequence forms indeed a martingale sequence where the associated filtration of the σ-algebras $\mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq \ldots \subseteq \mathcal{F}_n$ is defined so that $\mathcal{F}_i$ (for $i = 0, 1, \ldots, n$) is the σ-algebra that is generated by all the sub-sets of $n(1-R) \times n$ binary parity-check matrices whose first i columns are fixed. This martingale sequence satisfies $|X_i - X_{i-1}| \leq 1$ for $i = 1, \ldots, n$ (since if we reveal a new column of H, then the minimal number of linearly dependent columns can change by at most 1). Note that the RV X_0 is the expected minimum Hamming distance of the ensemble, and X_n is the minimum distance of a particular code from the ensemble (since once we revealed all the n columns of H, then the code is known exactly). Hence, by Azuma's inequality

$$\mathbb{P}(|d_{\min}(\mathcal{C}) - \mathbb{E}[d_{\min}(\mathcal{C})]| \geq \alpha\sqrt{n}) \leq 2 \exp\left(-\frac{\alpha^2}{2}\right), \ \forall\, \alpha > 0.$$

This leads to the following theorem:

Theorem 12 [The minimum distance of binary linear block codes]

Let $\mathcal{C}$ be chosen uniformly at random from the ensemble of binary linear block codes of length n and rate R. Then for every $\alpha > 0$, with probability at least $1 - 2\exp\left(-\frac{\alpha^2}{2}\right)$, the minimum distance of $\mathcal{C}$ is in the interval

$$[n\, h_2^{-1}(1-R) - \alpha\sqrt{n},\ n\, h_2^{-1}(1-R) + \alpha\sqrt{n}]$$

and it therefore concentrates around its expected value.

Note, however, that some well-known capacity-approaching families of binary linear block codes possess a minimum Hamming distance which grows sub-linearly with the block length n. For example, the class of parallel concatenated convolutional (turbo) codes was proved to have a minimum distance which grows at most like the logarithm of the interleaver length [26].

4.3.2 Performance of LDPC Codes under Iterative Message-Passing Decoding

In the following, we consider ensembles of binary LDPC codes. Following standard notation, let λ_i and ρ_i denote the fraction of edges attached, respectively, to variable and parity-check nodes of degree i. The LDPC code ensemble that is denoted by $\mathrm{LDPC}(n, \lambda, \rho)$ is characterized by the block length n of the codes, and the pair $\lambda(x) \triangleq \sum_i \lambda_i x^{i-1}$ and $\rho(x) \triangleq \sum_i \rho_i x^{i-1}$ which represent, respectively, the left and right degree distributions from the edge perspective.

The following theorem was proved in [49, Appendix C] based on Azuma's inequality:

Theorem 13 [Concentration of the bit error probability around the ensemble average]

Let $\mathcal{C}$, a code chosen uniformly at random from the ensemble $\mathrm{LDPC}(n, \lambda, \rho)$, be used for transmission over a memoryless binary-input output-symmetric (MBIOS) channel characterized by its L-density a_{MBIOS}. Assume that the decoder performs l iterations of message-passing decoding, and let $P_{\mathrm{b}}(\mathcal{C}, a_{\mathrm{MBIOS}}, l)$ denote the resulting bit error probability. Then, for every $\delta > 0$, there exists an $\alpha > 0$ where $\alpha = \alpha(\lambda, \rho, \delta, l)$ (*independent of the block length* n) such that

$$\mathbb{P}\left(|P_{\mathrm{b}}(\mathcal{C}, a_{\mathrm{MBIOS}}, l) - \mathbb{E}_{\mathrm{LDPC}(n,\lambda,\rho)}[P_{\mathrm{b}}(\mathcal{C}, a_{\mathrm{MBIOS}}, l)]| \geq \delta\right) \leq \exp(-\alpha n)$$

This theorem asserts that all except an exponentially (in the block length) small fraction of codes behave within an arbitrary small δ from the ensemble average (where δ is a positive number that can be chosen arbitrarily small). Therefore, assuming a sufficiently large block length, the ensemble average is a good indicator for the performance of individual codes, and it is therefore reasonable to focus on the design and analysis of capacity-approaching ensembles (via the density evolution technique). This theorem is proved in [49, pp. 487–490] based on Azuma's inequality.

4.4 A Tightened Large-Deviation Analysis for the Conditional Entropy of LDPC Ensembles

A large-deviation analysis of the conditional entropy for random ensembles of LDPC codes was introduced in [4, Theorem 1] and [8, Theorem 4]. The following theorem is proved in [8, Appendix I] based on Azuma's inequality:

Theorem 14 [Large deviations of the conditional entropy] Let $\mathcal{C}$ be chosen uniformly at random from the ensemble LDPC(n, λ, ρ). Assume that the transmission of the code $\mathcal{C}$ takes place over an MBIOS channel. Let $H(\mathbf{X}|\mathbf{Y})$ designate the conditional entropy of the transmitted codeword $\mathbf{X}$ given the received sequence $\mathbf{Y}$ from the channel. Then for any $\xi > 0$,

$$\mathbb{P}(|H(\mathbf{X}|\mathbf{Y}) - \mathbb{E}_{\mathrm{LDPC}(n,\lambda,\rho)}[H(\mathbf{X}|\mathbf{Y})]| \geq n\,\xi) \leq 2\exp(-nB\xi^2)$$

where $B \triangleq \frac{1}{2(d_{\mathrm{c}}^{\max}+1)^2(1-R_d)}$, $d_{\mathrm{c}}^{\max}$ is the maximal check-node degree, and R_{d} is the design rate of the ensemble.

The conditional entropy scales linearly with n, and this inequality considers deviations from the average which also scale linearly with n.

In the following, we revisit the proof of Theorem 14 in [8, Appendix I] in order to derive a tightened version of this bound. Based on this proof, let $\mathcal{G}$ be a bipartite graph which represents a code chosen uniformly at random from the ensemble LDPC(n, λ, ρ). Define the RV

$$Z = H_{\mathcal{G}}(\mathbf{X}|\mathbf{Y})$$

which forms the conditional entropy when the transmission takes place over an MBIOS channel whose transition probability is given by $P_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} p_{Y|X}(y_i|x_i)$ where $p_{Y|X}(y|1) = p_{Y|X}(-y|0)$. Fix an arbitrary order for the $m = n(1 - R_{\mathrm{d}})$ parity-check nodes where R_{d} forms the design rate of the LDPC code ensemble. Let $\{\mathcal{F}_t\}_{t\in\{0,1,\ldots,m\}}$ form a filtration of σ-algebras $\mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq \ldots \subseteq \mathcal{F}_m$ where $\mathcal{F}_t$ (for $t = 0, 1, \ldots, m$) is the σ-algebra that is generated by all the sub-sets of $m \times n$ parity-check matrices that are characterized by the pair of degree distributions (λ, ρ) and whose first t parity-check equations are fixed (for $t = 0$ nothing is fixed, and therefore $\mathcal{F}_0 = \{\emptyset, \Omega\}$ where $\emptyset$ denotes the empty set, and Ω is the whole sample space of $m \times n$ binary parity-check

matrices that are characterized by the pair of degree distributions (λ, ρ)). Accordingly, based on Remarks 13 and 14, let us define the following martingale sequence

$$Z_t = \mathbb{E}[Z|\mathcal{F}_t] \quad t \in \{0, 1, \ldots, m\}\,.$$

By construction, $Z_0 = \mathbb{E}[H_{\mathcal{G}}(\mathbf{X}|\mathbf{Y})]$ is the expected value of the conditional entropy for the LDPC code ensemble, and Z_m is the RV that is equal (a.s.) to the conditional entropy of the particular code from the ensemble (see Remark 14). Similarly to [8, Appendix I], we obtain upper bounds on the differences $|Z_{t+1} - Z_t|$ and then rely on Azuma's inequality in Theorem 11.

Without loss of generality, the parity-checks are ordered in [8, Appendix I] by increasing degree. Let $\mathbf{r} = (r_1, r_2, \ldots)$ be the set of parity-check degrees in ascending order, and Γ_i be the fraction of parity-check nodes of degree i. Hence, the first $m_1 = n(1 - R_{\mathrm{d}})\Gamma_{r_1}$ parity-check nodes are of degree r_1, the successive $m_2 = n(1 - R_{\mathrm{d}})\Gamma_{r_2}$ parity-check nodes are of degree r_2, and so on. The $(t+1)$th parity-check will therefore have a well defined degree, to be denoted by r. In order to avoid any further overlap with the proof in [8, Appendix I], we note that according to this proof

$$|Z_{t+1} - Z_t| \leq (r+1)\, H(\tilde{X}|\mathbf{Y}) \tag{4.1}$$

where $H(\tilde{X}|\mathbf{Y})$ is a RV which designates the conditional entropy of a parity-bit $\tilde{X} = X_{i_1} \oplus \ldots \oplus X_{i_r}$ (i.e., $\tilde{X}$ is equal to the modulo-2 sum of some r bits in the codeword $\mathbf{X}$) given the received sequence $\mathbf{Y}$ at the channel output. The proof in [8, Appendix I] was then completed by upper bounding the parity-check degree r by the maximal parity-check degree $d_{\mathrm{c}}^{\max}$, and also by upper bounding the conditional entropy of the parity-bit $\tilde{X}$ by 1. This gives

$$|Z_{t+1} - Z_t| \leq d_{\mathrm{c}}^{\max} + 1 \quad t = 0, 1, \ldots, m-1. \tag{4.2}$$

which then proves Theorem 14 from Azuma's inequality. Note that the d_i's in Theorem 11 are equal to $d_{\mathrm{c}}^{\max} + 1$, and n in Theorem 11 is replaced with the length $m = n(1 - R_{\mathrm{d}})$ of the martingale sequence $\{Z_t\}$ (that is equal to the number of the parity-check nodes in the graph).

In the continuation, we deviate from the proof in [8, Appendix I] in two respects:

- The first difference is related to the upper bound on the conditional entropy $H(\tilde{X}|\mathbf{Y})$ in (4.1) where $\tilde{X}$ is the modulo-2 sum of some r bits of the transmitted

codeword $\mathbf{X}$ given the channel output $\mathbf{Y}$. Instead of taking the most trivial upper bound that is equal to 1, as was done in [8, Appendix I], a simple upper bound on the conditional entropy is derived; this bound depends on the parity-check degree r and the channel capacity C (see Proposition 1).

- The second difference is minor, but it proves to be helpful for tightening the large-deviation inequality for LDPC code ensembles that are not right-regular (i.e., the case where the degrees of the parity-check nodes are not fixed to a certain value). Instead of upper bounding the term $r+1$ on the right-hand side of (4.1) with $d_{\mathrm{c}}^{\max}+1$, it is suggested to leave it as is since Azuma's inequality applies to the case where the bounded differences of the martingale sequence are not fixed (see Theorem 11), and since the number of the parity-check nodes of degree r is equal to $n(1-R_{\mathrm{d}})\Gamma_r$. The effect of this simple modification will be shown in Example 2.

The following upper bound is related to the first item above:

Proposition 1 Let $\mathcal{G}$ be a bipartite graph which corresponds to a binary linear block code whose transmission takes place over an MBIOS channel. Let $\mathbf{X}$ and $\mathbf{Y}$ designate the transmitted codeword and received sequence at the channel output. Let $\tilde{X} = X_{i_1} \oplus \ldots \oplus X_{i_r}$ be a parity-bit of some r code bits of $\mathbf{X}$. Then, the conditional entropy of $\tilde{X}$ given $\mathbf{Y}$ satisfies

$$H(\tilde{X}|\mathbf{Y}) \leq h_2\left(\frac{1-C^{\frac{r}{2}}}{2}\right). \tag{4.3}$$

Further, for a binary symmetric channel (BSC) or a binary erasure channel (BEC), this bound can be improved to

$$h_2\left(\frac{1-\left[1-2h_2^{-1}(1-C)\right]^r}{2}\right) \tag{4.4}$$

and

$$1-C^r \tag{4.5}$$

respectively, where h_2^{-1} in (4.4) designates the inverse of the binary entropy function on base 2.

Note that if the MBIOS channel is perfect (i.e., its capacity is $C = 1$ bit per channel use) then (4.3) holds with equality (where both sides of (4.3) are zero), whereas the trivial upper bound is 1.

Proof: Let us upper bound the conditional entropy $H(\tilde{X}\big|\mathbf{Y})$ with $H(\tilde{X}\big|Y_{i_1}, \ldots, Y_{i_r})$, where the latter conditioning refers to the intrinsic information for the bits $X_{i_1}, \ldots X_{i_r}$ which are used to calculate the parity-bit $\tilde{X}$. Then, from [15, Eq. (17) and Appendix I], the conditional entropy of the bit $\tilde{X}$ given the n-length received sequence $\mathbf{Y}$ satisfies the inequality

$$H(\tilde{X}\big|\mathbf{Y}) \leq 1 - \frac{1}{2\ln 2} \sum_{p=1}^{\infty} \frac{(g_p)^r}{p(2p-1)} \tag{4.6}$$

where (see [15, Eq. (19)])

$$g_p \triangleq \int_0^{\infty} a(l)(1+e^{-l}) \tanh^{2p}\left(\frac{l}{2}\right) dl, \quad \forall\, p \in \mathbb{N} \tag{4.7}$$

and $a(\cdot)$ denotes the symmetric *pdf* of the log-likelihood ratio at the output of the MBIOS channel, given that the channel input is equal to zero. From [15, Lemmas 4 and 5], it follows that

$$g_p \geq C^p, \quad \forall\, p \in \mathbb{N}.$$

Substituting this inequality in (4.6) gives that

$$\begin{aligned} H(\tilde{X}\big|\mathbf{Y}) &\leq 1 - \frac{1}{2\ln 2} \sum_{p=1}^{\infty} \frac{C^{pr}}{p(2p-1)} \\ &= h_2\left(\frac{1-C^{\frac{r}{2}}}{2}\right) \end{aligned} \tag{4.8}$$

where the last transition follows from the power series expansion of the binary entropy function where

$$h_2(x) = 1 - \frac{1}{2\ln 2} \sum_{p=1}^{\infty} \frac{(1-2x)^{2p}}{p(2p-1)}, \quad 0 \leq x \leq 1. \tag{4.9}$$

The tightened bound on the conditional entropy for the BSC is obtained from (4.6) and the equality

$$g_p = \left(1 - 2h_2^{-1}(1-C)\right)^{2p}, \quad \forall\, p \in \mathbb{N}$$

which holds for the BSC (see [15, Eq. (97)]). This replaces C on the right-hand side of (4.8) with $\left(1 - 2h_2^{-1}(1-C)\right)^2$, thus leading to the tightened bound in (4.4).

The tightened result for the BEC holds since from (4.7)

$$g_p = C, \quad \forall\, p \in \mathbb{N}$$

(see [15, Appendix II]), and a substitution of this equality in (4.6) gives (4.4) (note that $\sum_{p=1}^{\infty} \frac{1}{p(2p-1)} = 2\ln 2$). This completes the proof of Proposition 1.

From Proposition 1 and (4.1)

$$|Z_{t+1} - Z_t| \leq (r+1) h_2\left(\frac{1 - C^{\frac{r}{2}}}{2}\right) \tag{4.10}$$

with the corresponding two improvements for the BSC and BEC (where the second term on the right-hand side of (4.10) is replaced by (4.4) and (4.5), respectively). This improves the loosened bound $(d_{\mathrm{c}}^{\max} + 1)$ in [8, Appendix I].

From (4.10) and Theorem 11, we obtain the following tightened version of the large-deviation inequality in Theorem 14.

Theorem 15 [A first tightened large-deviation inequality for the conditional entropy] Let $\mathcal{C}$ be chosen uniformly at random from the ensemble LDPC(n, λ, ρ). Assume that the transmission of the code $\mathcal{C}$ takes place over an MBIOS channel. Let $H(\mathbf{X}|\mathbf{Y})$ designate the conditional entropy of the transmitted codeword $\mathbf{X}$ given the received sequence $\mathbf{Y}$ at the channel output. Then

$$\mathbb{P}(|H(\mathbf{X}|\mathbf{Y}) - \mathbb{E}_{\mathrm{LDPC}(n,\lambda,\rho)}[H(\mathbf{X}|\mathbf{Y})]| \geq n\,\xi) \leq 2\exp(-nB\xi^2)$$

for every $\xi > 0$, and

$$B \triangleq \frac{1}{2(1-R_{\mathrm{d}}) \sum_{i=1}^{d_{\mathrm{c}}^{\max}} (i+1)^2 \Gamma_i \left[h_2\left(\frac{1-C^{\frac{i}{2}}}{2}\right)\right]^2} \tag{4.11}$$

where $d_{\mathrm{c}}^{\max}$ is the maximal check-node degree, R_{d} is the design rate of the ensemble, and C is the channel capacity (in bits per channel use).

For the BSC and BEC, the parameter B can be improved (increased) to

$$B \triangleq \frac{1}{2(1-R_{\mathrm{d}}) \sum_{i=1}^{d_{\mathrm{c}}^{\max}} (i+1)^2 \Gamma_i \left[h_2\left(\frac{1-[1-2h_2^{-1}(1-C)]^i}{2}\right)\right]^2}$$

and

$$B \triangleq \frac{1}{2(1-R_{\mathrm{d}}) \sum_{i=1}^{d_{\mathrm{c}}^{\max}} (i+1)^2 \Gamma_i \left(1-C^i\right)^2} \tag{4.12}$$

respectively

Remark 15 From (4.11), Theorem 15 indeed yields a stronger large-deviation inequality than Theorem 14.

Remark 16 In the limit where $C \to 1$ bit per channel use, it follows from (4.11) that if $d_{\mathrm{c}}^{\max} < \infty$ then $B \to \infty$. This is in contrast to the value of B in Theorem 14 which does not depend on the channel capacity and is finite. Note that B should be indeed infinity for a perfect channel, and therefore Theorem 15 is tight in this case.

In the case where $d_{\mathrm{c}}^{\max}$ is not finite, we prove the following:

Lemma 5 If $d_{\mathrm{c}}^{\max} = \infty$ and $\rho'(1) < \infty$ then $B \to \infty$ in the limit where $C \to 1$.

Proof: See Appendix B.

This is in contrast to the value of B in Theorem 14 which vanishes when $d_{\mathrm{c}}^{\max} = \infty$, and therefore Theorem 14 is not informative in this case (see Example 2).

Example 1 [Comparison of Theorems 14 and 15 for right-regular LDPC code ensembles] In the following, we exemplify the improvement in the tightness of Theorem 15 for right-regular LDPC code ensembles. Consider the case where the communications takes place over a binary-input additive white Gaussian noise channel (BIAWGNC) or a BEC. Let us consider the $(2, 20)$ regular LDPC code ensemble whose design rate is equal to 0.900 bits per channel use. For a BEC, the threshold of the channel bit erasure probability under belief-propagation (BP) decoding is given by

$$p_{\mathrm{BP}} = \inf_{x \in (0,1]} \frac{x}{1-(1-x)^{19}} = 0.0531$$

which corresponds to a channel capacity of $C = 0.9469$ bits per channel use. For the BIAWGNC, the threshold under BP decoding is equal to $\sigma_{\mathrm{BP}} = 0.4156590$. From [49, Example 4.38] which expresses the capacity of the BIAWGNC in terms of the standard deviation σ of the Gaussian noise, the minimum capacity of a BIAWGNC over which it is possible to communicate with vanishing bit error probability under

BP decoding is $C = 0.9685$ bits per channel use. Accordingly, let us assume that for reliable communications on both channels, the capacity of the BEC and BIAWGNC is set to 0.98 bits per channel use.

Since the considered code ensembles is right-regular (e.g., the parity-check degree is fixed to $d_{\mathrm{c}} = 20$), then B in Theorem 15 is improved by a factor of

$$\frac{1}{\left[h_2\left(\frac{1-C^{\frac{d_{\mathrm{c}}}{2}}}{2}\right)\right]^2} = 5.134.$$

This implies that the inequality in Theorem 15 is satisfied with a block length that is 5.134 times shorter than the block length which corresponds to Theorem 14. For the BEC, the result is improved by a factor of

$$\frac{1}{\left(1 - C^{d_{\mathrm{c}}}\right)^2} = 9.051$$

due to the tightened value of B in (4.12) as compared to Theorem 14.

Example 2 [Comparison of Theorems 14 and 15 for a heavy-tail Poisson distribution (Tornado codes)] In the following, we compare Theorems 14 and 15 for Tornado LDPC code ensembles. This capacity-achieving sequence for the BEC refers to the heavy-tail Poisson distribution, and it was introduced in [6], [28, Section IV] (see also [49, Problem 3.20]). We rely in the following on the analysis in [15, Appendix VI].

Suppose that we wish to design Tornado code ensembles that achieve a fraction $1-\varepsilon$ of the capacity of a BEC under iterative message-passing decoding (where ε can be set arbitrarily small). Let p designate the bit erasure probability of the channel. The parity-check degree is Poisson distributed, and therefore the maximal degree of the parity-check nodes is infinity. Hence, $B = 0$ according to Theorem 14, and this theorem therefore is useless for the considered code ensemble. On the other hand,

from Theorem 15

$$\begin{aligned}
&\sum_i (i+1)^2 \Gamma_i \left[h_2 \left(\frac{1 - C^{\frac{i}{2}}}{2} \right) \right]^2 \\
&\overset{\text{(a)}}{\leq} \sum_i (i+1)^2 \Gamma_i \\
&\overset{\text{(b)}}{=} \frac{\sum_i \rho_i (i+2)}{\int_0^1 \rho(x) \, d\mathrm{x}} + 1 \\
&\overset{\text{(c)}}{=} (\rho'(1) + 3) d_{\mathrm{c}}^{\mathrm{avg}} + 1 \\
&\overset{\text{(d)}}{=} \left(\frac{\lambda'(0)\rho'(1)}{\lambda_2} + 3 \right) d_{\mathrm{c}}^{\mathrm{avg}} + 1 \\
&\overset{\text{(e)}}{\leq} \left(\frac{1}{p\lambda_2} + 3 \right) d_{\mathrm{c}}^{\mathrm{avg}} + 1 \\
&\overset{\text{(f)}}{=} O\left(\log^2 \left(\frac{1}{\varepsilon} \right) \right)
\end{aligned}$$

where inequality (a) holds since the binary entropy function on base 2 is bounded between zero and one, equality (b) holds since

$$\Gamma_i = \frac{\frac{\rho_i}{i}}{\int_0^1 \rho(x) \, \mathrm{d}x}$$

where Γ_i and ρ_i denote the fraction of parity-check nodes and the fraction of edges that are connected to parity-check nodes of degree i respectively (and also since $\sum_i \Gamma_i = 1$), equality (c) holds since

$$d_{\mathrm{c}}^{\mathrm{avg}} = \frac{1}{\int_0^1 \rho(x) \, \mathrm{d}x}$$

where $d_{\mathrm{c}}^{\mathrm{avg}}$ denotes the average parity-check node degree, equality (d) holds since $\lambda'(0) = \lambda_2$, inequality (e) is due to the stability condition for the BEC (where $p\lambda'(0)\rho'(1) < 1$ is a necessary condition for reliable communication on the BEC under BP decoding), and finally equality (f) follows from the analysis in [15, Appendix VI] (an upper bound on λ_2 is derived in [15, Eq. (120)], and the average parity-check node degree scales like $\log \frac{1}{\varepsilon}$). Hence, from the above chain of inequalities and (4.11), it follows that for a small gap to capacity, the parameter B in Theorem 15 scales (at

least) like

$$O\left(\frac{1}{\log^2\left(\frac{1}{\varepsilon}\right)}\right).$$

Theorem 15 is therefore useful for the large-deviation analysis of this LDPC code ensemble. As shown above, the parameter B in (4.11) tends to zero rather slowly as we let the fractional gap ε tend to zero (which therefore demonstrates a rather fast concentration in Theorem 15).

Example 3 This example forms a direct continuation of Example 1 for the $(n, d_{\mathrm{v}}, d_{\mathrm{c}})$ regular LDPC code ensembles where $d_{\mathrm{v}} = 2$ and $d_{\mathrm{c}} = 20$. With the settings in this example, Theorem 14 gives that

$$\begin{aligned}&\mathbb{P}(|H(\mathbf{X}|\mathbf{Y}) - \mathbb{E}_{\mathrm{LDPC}(n,\lambda,\rho)}[H(\mathbf{X}|\mathbf{Y})]| \geq n\,\xi)\\ &\leq 2\exp(-0.0113 n\xi^2)\end{aligned} \tag{4.13}$$

for every $\xi > 0$. As was mentioned already in Example 1, the exponential inequalities in Theorem 15 achieve an improvement in the exponent of Theorem 14 by factors 5.134 and 9.051 for the BIAWGNC and BEC, respectively. One therefore obtains via the inequalities in Theorem 15 that for every $\xi > 0$

$$\begin{aligned}&\mathbb{P}(|H(\mathbf{X}|\mathbf{Y}) - \mathbb{E}_{\mathrm{LDPC}(n,\lambda,\rho)}[H(\mathbf{X}|\mathbf{Y})]| \geq n\,\xi)\\ &\leq \begin{cases} 2\exp(-0.0580 n\xi^2) & \text{BIAWGNC}\\ 2\exp(-0.1023 n\xi^2), & \text{BEC}\end{cases}.\end{aligned} \tag{4.14}$$

4.5 Concentration for channels with ISI

Concentration analysis on the number of erroneous variable-to-check node messages for random ensembles of LDPC codes was introduced in [29] and [48] for memoryless channels. It was shown that the performance of an individual code from the ensemble concentrates around the expected (average) value over this ensemble when the block length of the code increases. Furthermore, the average behavior converges to the behavior of the cycle-free case. These results were later generalized in [3] for the case of channels with memory (i.e., for ISI channels), however no explicit expression for the concentration rate was provided. In this section, we revisit the proofs of [3,

Theorems 1 and 2] for the case of regular LDPC code ensembles in order to derive an explicit expression for the exponential rate that is related to the concentration inequality. It is then shown that particularizing the expression for memoryless channels provides a tightened concentration inequality as compared to [29] and [48].

4.5.1 The ISI Channel and its message-passing decoding

In the following, we briefly describe the ISI channel and the graph used for its message-passing decoding. For a detailed description, the reader is referred to [3]. Consider a binary discrete-time ISI channel with a finite memory length, to be denoted by I. The channel's output y_t at time $t \in \mathbb{Z}$ is given by the equality

$$y_t = \sum_{i=0}^{I} h_i x_{t-i} + n_t.$$

Where $x_t \in \{+1, -1\}$ is the channel's input, h_i is the channel's response and $n_t \sim N(0, \sigma^2)$ is an i.i.d. AWGN noise sequence. The information block of length k is coded using a regular $(n, d_{\mathrm{v}}, d_{\mathrm{c}})$ LDPC code, and the resulting n coded bits are converted to $x_t \in \{+1, -1\}$ before transmission over the channel. For decoding, we consider the windowed version of the "sum-product" algorithm when applied to ISI channels (see details in [3] and [20]). As in the memoryless case, this is a message passing algorithm. The variable-to-check and check-to-variable messages are computed as in the min-sum algorithm for the memoryless case. The difference is that a variable node's message from the trellis nodes is not only a function of the the channel output that corresponds to the considered symbol but also a function of $2W$ neighboring channel outputs and $2W$ neighboring variables nodes as illustrated in Fig. 4.1.

4.5.2 Concentration results for channels with ISI

It is proved in this sub-section that for a large n, a neighborhood of depth ℓ of a variable-to-check node message is tree-like with high probability. Using Azuma's inequality and the later result, it is shown that for most graphs and channel realizations, if $\mathbf{x}$ is the transmitted codeword, then the probability of a variable-to-check message being erroneous after ℓ rounds of message-passing decoding is highly concentrated around its expected value. This expected value is shown to converge to the value

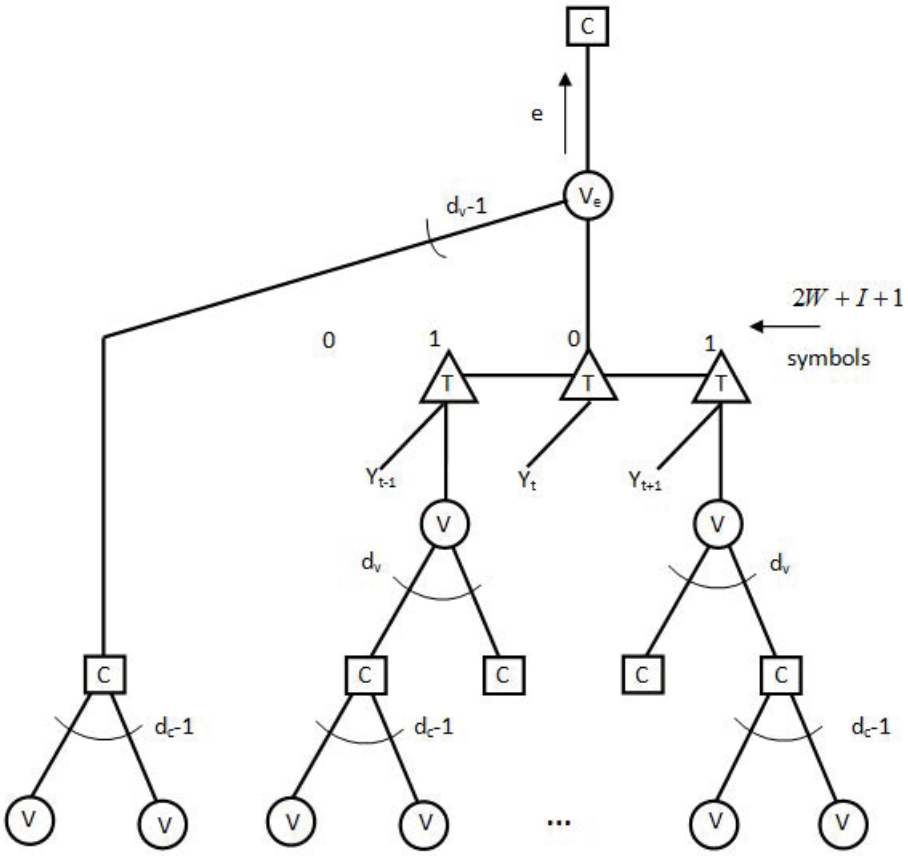

Figure 4.1: Message flow neighborhood of depth 1. The round, cubic, triangle nodes denote the variable, check and trellis nodes respectively. In this figure $(I, W, d_{\mathrm{v}}, d_{\mathrm{c}}) = (1, 1, 2, 3)$.

of $p^{(\ell)}(\mathbf{x})$ which corresponds to the cycle-free case. Also, we prove that if the transmitted sequence is i.u.d., then the probability highly concentrated around the value $p^{(\ell)}_{i.u.d.} \equiv \mathbb{E}[p^{(\ell)}(\mathbf{x})]$.

In the following theorems, we consider an ISI channel and windowed message-passing decoding algorithm, when the code graph is chosen uniformly at random from the ensemble of the graphs with variable and check node degree d_{v} and d_{c} respectively. Denote $\mathcal{N}^{(\ell)}_{\vec{e}}$ as the neighborhood of depth ℓ of an edge $\vec{e} = (\mathrm{v}, \mathrm{c})$ between a variable-to-check node. Let $N^{(\ell)}_{\mathrm{c}}$, $N^{(\ell)}_{\mathrm{v}}$ and $N^{(\ell)}_{e}$ denote the total number of check nodes, variable nodes and code related edges (variable-to-check node or check-to-variable node edges) respectively in this neighborhood. Similarly denote $N^{(\ell)}_{Y}$ as the number of variable-to-check node messages in the directed neighborhood of depth ℓ of a received value of the channel.

Theorem 16 [Probability of a neighborhood of depth ℓ of a variable-to-check node message to be tree-like for channels with ISI]

Define $P^{(\ell)}_{\bar{t}} \equiv \Pr\left\{\mathcal{N}^{(\ell)}_{\vec{e}} \text{ not a tree}\right\}$ as the probability that $\mathcal{N}^{(\ell)}_{\vec{e}}$ is not tree-like. Then, there exists a positive constant

$$\gamma(d_{\mathrm{v}}, d_{\mathrm{c}}, I, W, \ell) = {N^{(\ell)}_{\mathrm{v}}}^2 + \frac{d_{\mathrm{c}}}{d_{\mathrm{v}}} {N^{(\ell)}_{\mathrm{v}}}^2 \tag{4.15}$$

such that

$$P^{(\ell)}_{\bar{t}} \leq \frac{\gamma}{n}.$$

Proof: This proof follows from the proof in [48] and extends it to the case of channels with ISI. Consider a neighborhood $\mathcal{N}^{(\ell)}_{\vec{e}}$ of fixed depth ℓ . Note that at each level the graph expands by factor $\alpha \equiv (d_{\mathrm{v}} - 1 + 2Wd_{\mathrm{v}})(d_{\mathrm{c}} - 1)$, therefore there are in total

$$N^{(\ell)}_{\mathrm{v}} = 1 + [(d_{\mathrm{v}} - 1)(d_{\mathrm{c}} - 1) + 2W(1 + d_{\mathrm{v}}(d_{\mathrm{c}} - 1))] \sum_{i=0}^{\ell-1} \alpha^i$$

variable nodes and

$$N^{(\ell)}_{\mathrm{c}} = 1 + (d_{\mathrm{v}} - 1 + 2Wd_{\mathrm{v}}) \sum_{i=0}^{\ell-1} \alpha^i$$

check nodes in this neighborhood.

In order to lower bound $P_{\bar{t}}^{(\ell)}$ we can upper bound $P_t^{(\ell)} = 1 - P_{\bar{t}}^{(\ell)}$. This is done by factorizing $P_t^{(\ell)}$ as

$$P_t^{(\ell)} = \Pr\left\{\mathcal{N}_{\vec{e}}^{(0)} \text{ is tree}\right\} \prod_{\ell^*=0}^{\ell-1} \Pr\left\{\mathcal{N}_{\vec{e}}^{(\ell^*+1)} \text{ is tree} | \mathcal{N}_{\vec{e}}^{(\ell^*)} \text{ is tree}\right\} \tag{4.16}$$

and bounding each factor. For $\ell^* = 0$ we have a single edge which is a tree, therefore $\Pr\left\{\mathcal{N}_{\vec{e}}^{(0)} \text{ is tree}\right\} = 1$. To bound $\Pr\left\{\mathcal{N}_{\vec{e}}^{(\ell^*+1)} \text{ is tree} | \mathcal{N}_{\vec{e}}^{(\ell^*)} \text{ is tree}\right\}$ we assume that $\mathcal{N}_{\vec{e}}^{(\ell^*)}$ is tree-like and reveal the code related edges (variable-to-check node or vice versa, as opposed to the channel related edges which are predetermined) one at a time. If in this process (of revealing the ℓ^*+1-th level of the tree) no loops are created then $\mathcal{N}_{\vec{e}}^{(\ell^*+1)}$ is also a tree. We start by revealing the leaves of a variable node . As opposed to the case with no ISI, where each variable node has only $d_{\text{v}} - 1$ direct paths to check nodes from the next level, here also $2Wd_{\text{v}}$ indirect paths through trellis nodes exist (i.e, variable-trellis-variable-check). Since the edges connected to a trellis node are predetermined, then an indirect path requires the revelation of a single variable-to-check node edge. Assume that k additional edges have been revealed at this stage without creating a loop. The next revealed edge is chosen among $(md_{\text{c}} - k - N_{\text{c}}^{(\ell^*)})$ edges and it does not create a loop if it is connected to one of the $(m - k - N_{\text{c}}^{(\ell^*)})$ un-explored check nodes. Since each un-explored check node has d_{c} edges, then the probability for not creating a loop is given by $\frac{(m-k-N_{\text{c}}^{(\ell^*)})d_{\text{c}}}{md_{\text{c}}-k-N_{\text{c}}^{(\ell^*)}}$. For large n we have

$$\begin{aligned} \frac{(m-k-N_{\text{c}}^{(\ell^*)})d_{\text{c}}}{md_{\text{c}}-k-N_{\text{c}}^{(\ell^*)}} &= \frac{(N_{\text{c}}^{(\ell^*)}+k)(d_{\text{c}}-1)}{md_{\text{c}}-k-N_{\text{c}}^{(\ell^*)}} \\ &\geq 1 - \frac{N_{\text{c}}^{(\ell)}}{m}. \end{aligned} \tag{4.17}$$

Since we have $N_{\text{c}}^{(\ell^*+1)} - N_{\text{c}}^{(\ell^*)}$ edges to reveal (one for each check node), then the probability that revealing all the leaves does not create a loop, given $\mathcal{N}_{\vec{e}}^{(\ell^*)}$ is tree-like is lower bounded by $\left(1 - \frac{N_{\text{c}}^{(\ell)}}{m}\right)^{N_{\text{c}}^{(\ell^*+1)} - N_{\text{c}}^{(\ell^*)}}$. Next, we reveal the outgoing edges of the check node leaves one at a time (here only d_{c} direct paths exist, as in the case without ISI). Assuming k variable nodes have been revealed without creating a loop, then the probability that the next revealed edge does no create a loop is $\frac{(n-k-N_{\text{v}}^{(\ell^*)})d_{\text{v}}}{nd_{\text{v}}-k-N_{\text{v}}^{(\ell^*)}}$.

For large n we have

$$\begin{aligned}\frac{(n-k-N_{\mathrm{v}}^{(\ell^*)})d_{\mathrm{v}}}{nd_{\mathrm{v}}-k-N_{\mathrm{v}}^{(\ell^*)}} &= \frac{(N_{\mathrm{v}}^{(\ell^*)}+k)(d_{\mathrm{v}}-1)}{nd_{\mathrm{v}}-k-N_{\mathrm{c}}^{(\ell^*)}} \\ &\geq 1-\frac{N_{\mathrm{v}}^{(\ell)}}{n}.\end{aligned} \tag{4.18}$$

Since we have $N_{\mathrm{v}}^{(\ell^*+1)}-N_{\mathrm{v}}^{(\ell^*)}$ edges to reveal (one for each variable node), then the probability that revealing all the leaves does not create loop, given the neighborhood is tree-like so far is lower bounded by $\left(1-\frac{N_{\mathrm{v}}^{(\ell)}}{n}\right)^{N_{\mathrm{v}}^{(\ell^*+1)}-N_{\mathrm{v}}^{(\ell^*)}}$. Combining (4.16), (4.17), (4.18) and $P_t^{(\ell)}=1-P_{\bar{t}}^{(\ell)}$ we have

$$P_{\bar{t}}^{(\ell)} \leq 1-\left(1-\frac{N_{\mathrm{v}}^{(\ell)}}{n}\right)^{N_{\mathrm{v}}^{(\ell)}}\left(1-\frac{N_{\mathrm{c}}^{(\ell)}}{m}\right)^{N_{\mathrm{c}}^{(\ell)}}.$$

Thus, for n sufficiently large

$$P_{\bar{t}}^{(\ell)} \leq \frac{{N_{\mathrm{v}}^{(\ell)}}^2+\frac{d_{\mathrm{c}}}{d_{\mathrm{v}}}{N_{\mathrm{c}}^{(\ell)}}^2}{n} \equiv \frac{\gamma}{n}$$

Remark 17 We note that if we the degenerate the expression for γ in (4.15) by setting $I=W=0$, we get the exact same expression given in [48] for channels without memory.

Theorem 17 [Concentration of the number of erroneous variable-to-check node messages for channels with ISI]

Let $\mathbf{x}$ be the transmitted codeword. Let $Z^{(\ell)}(\mathbf{x})$ be the number of erroneous variable-to-check node messages after ℓ rounds of the windowed message-passing decoding algorithm when the code graph is chosen uniformly at random from the ensemble of the graphs with variable and check node degree d_{v} and d_{c} respectively. Let $p^{(\ell)}(\mathbf{x})$ be the expected fraction of incorrect messages passed along an edge with a tree-like directed neighborhood of depth ℓ. Then, there exist positive constants $\beta(d_{\mathrm{v}},d_{\mathrm{c}},I,W,\ell)=\frac{d_{\mathrm{v}}^2}{8\left(16d_{\mathrm{v}}(N_e^{(\ell)})^2+(N_Y^{(\ell)})^2\right)}$ and $\gamma(d_{\mathrm{v}},d_{\mathrm{c}},I,W,\ell)={N_{\mathrm{v}}^{(\ell)}}^2+\frac{d_{\mathrm{c}}}{d_{\mathrm{v}}}{N_{\mathrm{c}}^{(\ell)}}^2$ such that

[**Concentration around expectation**] For any $\epsilon>0$ we have

$$\Pr\left\{\left|\frac{Z^{(\ell)}(\mathbf{x})}{nd_{\mathrm{v}}}-\frac{\mathbb{E}[Z^{(\ell)}(\mathbf{x})]}{nd_{\mathrm{v}}}\right|>\epsilon/2\right\} \leq 2e^{-\beta\epsilon^2 n} \tag{4.19}$$

[**Convergence to cycle-free case**] For any $\epsilon > 0$ and $n > \frac{2\gamma}{\epsilon}$ we have

$$\left| \frac{\mathbb{E}[Z^{(\ell)}(\mathbf{x})]}{nd_{\mathrm{v}}} - p^{(\ell)}(\mathbf{x}) \right| < \epsilon/2 \tag{4.20}$$

[**Concentration around cycle-free case**] For any $\epsilon > 0$ and $n > \frac{2\gamma}{\epsilon}$ we have

$$\Pr\left\{ \left| \frac{Z^{(\ell)}(\mathbf{x})}{nd_{\mathrm{v}}} - p^{(\ell)}(\mathbf{x}) \right| > \epsilon \right\} \leq 2e^{-\beta\epsilon^2 n} \tag{4.21}$$

Proof: First note that the following inequality holds

$$\begin{aligned} \Pr\left\{ \left| \frac{Z^{(\ell)}(\mathbf{x})}{nd_{\mathrm{v}}} - p^{(\ell)}(\mathbf{x}) \right| > \epsilon \right\} &\leq \Pr\left\{ \left| \frac{Z^{(\ell)}(\mathbf{x})}{nd_{\mathrm{v}}} - \frac{\mathbb{E}[Z^{(\ell)}(\mathbf{x})]}{nd_{\mathrm{v}}} \right| > \epsilon/2 \right\} \\ &+ \Pr\left\{ \left| \frac{\mathbb{E}[Z^{(\ell)}(\mathbf{x})]}{nd_{\mathrm{v}}} - p^{(\ell)}(\mathbf{x}) \right| > \epsilon/2 \right\}. \end{aligned} \tag{4.22}$$

If inequality (4.20) holds, then $\Pr\left\{ \left| \frac{Z^{(\ell)}(\mathbf{x})}{nd_{\mathrm{v}}} - p^{(\ell)}(\mathbf{x}) \right| > \epsilon/2 \right\} = 0$, therefore using (4.22) we deduce that (4.21) follows from (4.19) and (4.20). We start by proving (4.19). For a deterministic sequence $\mathbf{x}$, the random variable $Z^{(\ell)}(\mathbf{x})$ denotes the number of incorrect variable-to-check node messages among all nd_{v} variable-to-check node messages passed in the ℓth iteration for a particular graph $\mathcal{G}$ and decoder's input $\mathbf{y}$. Let us form a Doob's martingale by first exposing the nd_{v} edges of the graph one by one and then exposing the n received values y_i one by one. For $i = 0, ...n(d_{\mathrm{v}} + 1)$, define the RV $\tilde{Z}_i = \mathbb{E}[Z^{(\ell)}(\mathbf{x})|a_1, ...a_i]$. Where the sequence a is the sequence of the nd_{v} variable-to-check node edges of the graph followed by the sequence of the n received values. Note that it is a martingale sequence where $\tilde{Z}_0 = \mathbb{E}[Z^{(\ell)}(\mathbf{x})]$ and $\tilde{Z}_{n(d_{\mathrm{v}}+1)} = Z^{(\ell)}(\mathbf{x})$. We can use Azuma's inequality if we can bound the sequence of differences $|\tilde{Z}_{i+1} - \tilde{Z}_i| \leq d_i$.

We now consider the effect of exposing an edge of the graph. Consider two graphs $\mathcal{G}$ and $\tilde{\mathcal{G}}$ whose edges are identical except for an exchange of the endpoint of two edges. A variable-to-check message is affected by this change if one (or both) of the edges is in its directed neighborhood of depth ℓ.

Consider a neighborhood of depth ℓ of a variable-to-check node message. Since at each level the graph expands by factor $\alpha \equiv (d_{\mathrm{v}} - 1 + 2Wd_{\mathrm{v}})(d_{\mathrm{c}} - 1)$ then there are,

in total

$$N_e^{(\ell)} = 1 + d_{\mathrm{c}}(d_{\mathrm{v}} - 1 + 2Wd_{\mathrm{v}}) \sum_{i=0}^{\ell-1} \alpha^i$$

edges related to the code structure (variable-to-check node edges or vice versa) in the neighborhood $\mathcal{N}_{(\ell)}^{\vec{e}}$. By symmetry the two edges can affect at most $4N_e^{(\ell)}$ neighborhoods (Alternatively we could directly sum the number of variable-to-check node edges in a neighborhood of a variable-to-check node edge and in a neighborhood of a check-to-variable node edge). The change in the number of incorrect variable-to-check node messages is bounded by the case that each change in the neighborhood of a message introduces an error. In a similar manner, when we reveal a received value, then variable-to-check node messages whose directed neighborhood include that channel input can be affected. We consider a neighborhood of depth ℓ of a received value. By counting, it can be shown that this neighborhood includes

$$N_Y^{(\ell)} = d_{\mathrm{v}}(2W + 1) \sum_{i=0}^{\ell-1} \alpha^i$$

variable-to-check node edges. Therefore a change in a received value can affect up to $N_Y^{(\ell)}$ variable-to-check node messages. We conclude that $d_i \leq 4N_e^{(\ell)}$ for the first $d_{\mathrm{v}}n$ exposures and $d_i \leq N_Y^{(\ell)}$ for the last n exposures. By applying Azuma's inequality we get

$$\Pr\left\{ \left| \frac{Z^{(\ell)}(\mathbf{x})}{nd_{\mathrm{v}}} - \frac{\mathbb{E}[Z^{(\ell)}(\mathbf{x})]}{nd_{\mathrm{v}}} \right| > \epsilon/2 \right\} \leq e^{-\frac{(nd_{\mathrm{v}}\epsilon/2)^2}{2[nd_{\mathrm{v}}(16N_e^{(\ell)})^2 + n(N_Y^{(\ell)})^2]}}$$

By comparing the result to (4.19), we get an expression for β

$$\frac{1}{\beta} = 8\left(16d_{\mathrm{v}}(N_e^{(\ell)})^2 + (N_Y^{(\ell)})^2\right)/d_{\mathrm{v}}^2$$

Next, we prove inequality (4.20), again it is adopted from [3] and [48]. Let $\mathbb{E}[Z_i^{(\ell)}(\mathbf{x})], i \in [nd_{\mathrm{v}}]$ be the expected number of incorrect messages passed along edge $\overrightarrow{e}_i$, where the average is over all graphs and all received values. Then by linearity of expectation and by symmetry

$$\mathbb{E}[Z^{(\ell)}(\mathbf{x})] = \sum_{i \in [nd_{\mathrm{v}}]} \mathbb{E}[Z_i^{(\ell)}(\mathbf{x})] = nd_{\mathrm{v}}\mathbb{E}[Z_1^{(\ell)}(\mathbf{x})]. \tag{4.23}$$

Furthermore

$$\mathbb{E}[Z_1^{(\ell)}(\mathbf{x})] = \mathbb{E}[Z_1^{(\ell)}(\mathbf{x})|\mathcal{N}_{\vec{e}}^{(\ell)} \text{ is tree}]P_t^{(\ell)} + \mathbb{E}[Z_1^{(\ell)}(\mathbf{x})|\mathcal{N}_{\vec{e}}^{(\ell)} \text{ not a tree}]P_{\bar{t}}^{(\ell)}.$$

As shown in Theorem 16, $P_{\bar{t}}^{(\ell)} \leq \frac{\gamma}{n}$ where γ is a positive constant independent of n. Furthermore, we have $\mathbb{E}[Z_1^{(\ell)}(\mathbf{x})|\text{neighborhood is tree}] = p^{(\ell)}(\mathbf{x})$ and by definition $0 \leq \mathbb{E}[Z_1^{(\ell)}(\mathbf{x})|\text{neighborhood not a tree}] \leq 1$. Hence

$$\begin{aligned} \mathbb{E}[Z_1^{(\ell)}(\mathbf{x})] &\leq (1 - P_{\bar{t}}^{(\ell)})p^{(\ell)}(\mathbf{x}) + P_{\bar{t}}^{(\ell)} \leq p^{(\ell)}(\mathbf{x}) + P_{\bar{t}}^{(\ell)} \\ \mathbb{E}[Z_1^{(\ell)}(\mathbf{x})] &\geq (1 - P_{\bar{t}}^{(\ell)})p^{(\ell)}(\mathbf{x}) \geq p^{(\ell)}(\mathbf{x}) - P_{\bar{t}}^{(\ell)}. \end{aligned} \tag{4.24}$$

Using (4.23), (4.24) and $P_{\bar{t}}^{(\ell)} \leq \frac{\gamma}{n}$ we get that

$$\left| \frac{\mathbb{E}[Z^{(\ell)}(\mathbf{x})]}{nd_{\text{v}}} - p^{(\ell)}(\mathbf{x}) \right| \leq P_{\bar{t}}^{(\ell)} \leq \frac{\gamma}{n}. \tag{4.25}$$

If we assume that $n > \frac{2\gamma}{\epsilon}$ then from (4.25), it follows that (4.20) holds. Since (4.19) and (4.20) hold then (4.21) is proved.

Discussion 1 The concentration result proved above is a generalization of the results given in [48] for the memoryless case. One can degenerate the expression $\frac{1}{\beta} = 8\left(16d_{\text{v}}(N_e^{(\ell)})^2 + (N_Y^{(\ell)})^2\right)/d_{\text{v}}^2$ to the memoryless case by setting $W = 0$ and $I = 0$. Since we used exact expressions for $N_e^{(\ell)}$ and $N_Y^{(\ell)}$ in the proof, we can expect a tighter bound as compared to the earlier result $\frac{1}{\beta_{\text{old}}} = 544 d_{\text{v}}^{2\ell-1} d_{\text{c}}^{2\ell}$ given in [48]. For example for $(d_{\text{v}}, d_{\text{c}}, \ell) = (3, 4, 10)$ we get an improvement by a factor of about 1 million. However even with this improved expression, the required size of n according to our proof can be absurdly large. This is because the proof is very pessimistic. We assume that any change in an edge or the decoder's input will introduce an error in every message it affects. This is especially pessimistic if large ℓ is considered, since as ℓ grows each message is a function of many edges and received values (since the neighborhood grows with ℓ). However in practice, the probability that changing a single edge or input will change the message is close to zero for long codes.

Theorem 18 Let $\mathbf{x}$ be a random sequence of i.u.d. binary variables $x_1, x_2 .... x_n$. Let $Z^{(\ell)}(\mathbf{x})$ be the number of erroneous variable-to-check messages after ℓ rounds of the

windowed message-passing decoding algorithm when the code graph is chosen uniformly at random from the ensemble of the graphs with variable and check node degree $d_{\rm v}$ and $d_{\rm c}$ respectively. Let $p^{(\ell)}_{i.u.d.} \equiv \mathbb{E}[p^{(\ell)}(\mathbf{x})]$ be the expected fraction of incorrect messages passed along an edge with a tree-like directed neighborhood of depth ℓ. Then, there exist positive constants $\beta' = \beta(d_{\rm v}, d_{\rm c}, I, W, \ell)$ and $\gamma = \gamma(d_{\rm v}, d_{\rm c}, I, W, \ell)$ such that for any $\epsilon > 0$ and $n > \frac{2\gamma}{\epsilon}$ we have

$$\Pr\left\{\left|\frac{Z^{(\ell)}(\mathbf{x})}{nd_{\rm v}} - p^{(\ell)}_{i.u.d.}\right| > \epsilon\right\} \leq 4e^{-\beta'\epsilon^2 n}. \tag{4.26}$$

Furthermore, $p^{(\ell)}_{i.u.d.}$ is equal to the error probability when all neighborhood types are equally probable.

Proof: The proof follows closely to the proof presented in [3]. First, note that the following chain of inequalities hold

$$\begin{aligned}
\Pr\left\{\left|\frac{Z^{(\ell)}(\mathbf{x})}{nd_{\rm v}} - p^{(\ell)}_{i.u.d.}\right| > \epsilon\right\} &= \sum_{j=1}^{2^n} 2^{-n} \Pr\left\{\left|\frac{Z^{(\ell)}(x_j)}{nd_{\rm v}} - p^{(\ell)}_{i.u.d.}\right| > \epsilon\right\} \\
&\leq \sum_{j=1}^{2^n} 2^{-n} \Pr\left\{\left|\frac{Z^{(\ell)}(x_j)}{nd_{\rm v}} - p^{(\ell)}(x_j)\right| > \epsilon/2\right\} \\
&\quad + \sum_{j=1}^{2^n} 2^{-n} \Pr\left\{\left|p^{(\ell)}(x_j) - p^{(\ell)}_{i.u.d.}\right| > \epsilon/2\right\} \\
&\leq \sum_{j=1}^{2^n} 2^{-n} \cdot 2e^{-\beta\epsilon^2 n/4} + \Pr\left\{\left|p^{(\ell)}(\mathbf{x}) - p^{(\ell)}_{i.u.d.}\right| > \epsilon/2\right\} \\
&= 2e^{-\beta\epsilon^2 n/4} + \Pr\left\{\left|p^{(\ell)}(\mathbf{x}) - p^{(\ell)}_{i.u.d.}\right| > \epsilon/2\right\}.
\end{aligned} \tag{4.27}$$

To bound the second term in the last line we shall use Azuma's inequality. Let us form a Doob's martingale by exposing the n received symbols one by one. For $t = 1, ..., n$, define the RV $M_t = \mathbb{E}[p^{(\ell)}(\mathbf{x})|x_1, x_2, ..., x_t]$. Note that $M_0 = \mathbb{E}[p^{(\ell)}(\mathbf{x})] = p^{(\ell)}_{i.u.d.}$ and $M_n = \mathbb{E}[p^{(\ell)}(\mathbf{x})|x_1, x_2, ..., x_n] = p^{(\ell)}(\mathbf{x})$. In order to use Azuma's inequality we shall show that the sequence of differences is bounded $|M_{t+1} - M_t| \leq d_t$. Since the channel has ISI of degree I, then exposing a single channel input affects I channel outputs (which are the received values for the decoder). A variable-to-check node

message is affected only if one of the affected received values are in its neighborhood. Therefore, changing a channel input can affect at most $IN_Y^{(\ell)}$ variable-to-check node messages among the $nd_{\rm v}$ messages in the graph. Thus $|M_{t+1} - M_t| \leq \frac{IN_Y^{(\ell)}}{nd_{\rm v}}$, and by using Azuma's inequality we have

$$\Pr\left\{\left|p^{(\ell)}(\mathbf{x}) - p_{i.u.d.}^{(\ell)}\right| > \epsilon/2\right\} \leq 2e^{-\delta\epsilon^2 n} \tag{4.28}$$

where $\delta = \frac{1}{8}\left(\frac{d_{\rm v}}{IN_Y^{\ell}}\right)^2$. Combining (4.28), (4.28) and comparing it to (4.26) gives that $\beta' = \min(\beta, \delta)$.

Next, we get an expression for $p_{i.u.d.}^{(\ell)}$ and show it is equal to the error probability when all neighborhood types are equally probable. This part of the proof is shown in [3], and is given for the completeness of the proof. In Fig. 4.1, a depth 1 message-flow neighborhood is shown. The row of bits "0101" given above the trellis section represent the binary symbols of the codeword $\mathbf{x}$ corresponding to the trellis nodes that influence the message flow. Since the channel has ISI memory of length I, there are $2W + I + 1$ binary symbols of that influence the message flow. We call this sequence of bits a neighborhood type. For example, in Fig. 4.1 the neighborhood type is $\theta = [0101]$. We expand this definition to a depth ℓ neighborhood by cascading the bits of each sub-neighborhood of depth ℓ. Since at each level, the graph expends by factor $\alpha \equiv (d_{\rm v} - 1 + 2Wd_{\rm v})(d_{\rm c} - 1)$ then there are exactly $2^{N(\ell)}$ possible types of message flow neighborhoods of depth ℓ, where

$$N(\ell) = (2W + I + 1)\sum_{i=0}^{\ell-1}\alpha^i = (2W + I + 1)\frac{\alpha^\ell - 1}{\alpha - 1}.$$

We can now define

$$\pi_{\underline{\theta}}^{(\ell)} = \Pr\left(\text{tree delivers incorrect message}|\text{tree type } \theta\right)$$

and

$$P(\underline{\theta}|\mathbf{x}) = \Pr\left(\text{tree type } \theta|\text{transmitted sequence} = \mathbf{x}\right)$$

Therefore we can express $p^{(\ell)}(\mathbf{x})$ as

$$p^{(\ell)}(\mathbf{x}) = \sum_{i=0}^{2^{N(\ell)}} \pi_{\underline{\theta}_i}^{(\ell)} \Pr(\underline{\theta}_i|\mathbf{x}).$$

Next, recognize that if $\mathbf{x}$ is an i.u.d. sequence, all neighborhood types are equally probable, i.e. $\Pr(\underline{\theta}|\mathbf{x}) = 2^{-N(\ell)}$. Using this we have

$$\begin{aligned}
\mathbb{E}[p^{(\ell)}(\mathbf{x})] &= \sum_{j=0}^{2^n} 2^{-n} p^{(\ell)}(x_j) \\
&= \sum_{j=0}^{2^n} 2^{-n} \sum_{i=0}^{2^{N(\ell)}} \pi_{\underline{\theta}_i}^{(\ell)} \Pr(\underline{\theta}_i|x_j) \\
&= \sum_{j=0}^{2^{N(\ell)}} \pi_{\underline{\theta}_i}^{(\ell)} \sum_{i=0}^{2^n} 2^{-n} \Pr(\underline{\theta}_i|x_j) \\
&= \sum_{i=0}^{2^{N(\ell)}} \pi_{\underline{\theta}_i}^{(\ell)} \Pr(\underline{\theta}_i|\mathbf{x}) \\
&= \sum_{i=0}^{2^{N(\ell)}} \pi_{\underline{\theta}_i}^{(\ell)} 2^{-N(\ell)}
\end{aligned}$$

The last term is equal to the error probability when all neighborhood types are equally probable. Since $\mathbb{E}[p^{(\ell)}(\mathbf{x})] = p_{i.u.d.}^{(\ell)}$ the theorem is proved.

Chapter 5

Summary and Conclusions

5.1 Contribution of the Thesis and conclusions

This thesis is focused on aspects of convex optimization and concentration in coding. In the following, we summarize the main contributions and conclusions in the thesis. The results are organized according to the three main subjects discussed.

5.1.1 Complexity analysis of convex optimization

In chapter 2, we consider the complexity of the interior-point method (IPM) for solving convex optimization problems. The analysis is limited to problems whose objective function is self-concordant.

We proved tightened bounds on the number of Newton iterations required to solve UCSCM, ECSCM and IESCM problems. The bounds where given for backtracking line-search and for two choices of predetermined step-size t. The bound given for backtracking line-search is 10-100 times tighter compared to previous bounds considering this line-search algorithm. The predetermined step-size was chosen to optimize the bound on the number of Newton iterations, resulting in a further improvement (about 5-10 times smaller scaling factor for $f(x^{(0)}) - p^*$) compared to the tightened bound given for backtracking line-search. We note that the bounds for predetermined step-size can be used as a bound for Newton's method with exact line-search since the pre-determined step size is a sub-optimal choice of t.

In the derivation of the described bounds, we have also proved several important

lemmas and Theorems. Lemma 1 and Theorem 7 establishes a rigid theoretical proof to the extension of the upper bound on the number of Newton iteration from unconstrained problems to equality and inequality constrained problems. Therefore, if a bound for an UCSCM problem is known and certain conditions are met, then the bound can be extended to ECSCM and IESCM problems. Lemma 2 provides a recursive bound on the Newton decrement. This lemma extends a previous bound [45] that was valid only for a full Newton step (i.e., $t = 1$), to a bound with the step-size t given as a free parameter.

Although tight compared to previous bounds, numerical results show that the tightened bounds still give a very conservative estimate for the number of Newton iterations. Using numerical analysis, we point out the two main reasons that keep the improved bounds from being tight :

1. The usage a of a global bound on the Newton decrement during the damped phase.

2. An artificial constraint on the bounded step-size caused by the fact that the bounds used have a domain of $t\lambda < 1$, whereas in practice $t\lambda > 1$ in the damped phase.

Furthermore, numerical results show that although the predetermined choice of t gives a tighter bound as compared to the backtracking bound, numerically, the algorithm is less efficient. We conclude that the predetermined choice optimizes the bound on the number of Newton iterations but not necessarily the practical efficiency of the solver. Thus, the optimized value of t is an artifact of the bounding method used.

5.1.2 Complexity analysis of IPM-based LP decoders

In chapter 3, we apply the interior-point method to the relaxed MLD problem. We provide an IPM-based LP decoder similar to the decoder given in [50],[51]. Based on the complexity bounds derived in previous chapter, the number of Newton iterations for the LP decoder is bounded. Similar analysis was presented in [50], however that analysis was valid only for row regular codes using the backtracking line-search whereas this analysis applies to general non-regular codes with backtracking, exact or predetermined line-search. Moreover, since tightened bounds from previous chapter

were used, the new bounds are much tighter compared to previous bounds. Using the derived analytic bound, we obtain a set of optimization parameters that optimize the derived bound. These optimized parameters provide an optimized bound that increases much slower with respect to the code's parameters. As an example, the original bound scales like $O(M \ln M)$ whereas the optimized bound scales only as $O(\sqrt{M} \ln M)$. Next, we analyze the behavior of the derived bounds as a function of the code and channel parameters. If the optimized bound is considered, the bound on the number of Newton iteration shows several interesting behaviors.

- The bound scales like $O(\sqrt{M} \ln M)$, where M denotes the number of inequalities required to describe the polytope. The number of inequalities is a measure of the complexity of the polytope. As the number of inequalities grow larger, the polytope is getting more complex, better representing the code, thus better decoding performance is maintained. This result express the trade-off between decoding performance (represented by the number of inequalities) and the decoding complexity (represented by the number of Newton iterations times the complexity per iteration).

- When considering the time complexity of the code, for general linear codes the bound scales like $O(2^n)$, whereas for LDPC codes the time complexity is only $O(n^{1.5} \ln n)$. Therefore the decoder is mainly suitable for LDPC codes.

- The complexity of the bound is exponential with respect to the check-node degree d_{c}. Therefore, the complexity remains low only for LDPC codes with low check-node degree.

The last two properties suggest that the proposed LP decoder has long running times for general linear codes with high check-node degree. On the other hand, for LDPC codes, the bound predicts relatively low complexity, making it suitable for this family of linear codes. For high-density codes (where $d_{\mathrm{c}}^{\max}$ scales like $O(n)$), alternative polytopes [21],[25] provide lower complexity. For these polytopes, we show that the bound on the number of iterations scales like $O(n^{1.5} \ln n)$ with complexity that scales like $O(n^{4.5} \ln n)$.

Finally, we compare the new tightened bound, to a previous bound given in [50]. For the particular code and channel considered, it is shown that an improvement by

factor 10 is achieved. However, even with this improved bound, the upper bound gives a very conservative estimate of the number of Newton iterations that does not reflect average case behaviors. However, if we use the optimized bound with predetermined step-size the resulting bound is a more reasonable estimate for the number of Newton iterations.

5.1.3 Concentration of measures in LDPC code ensembles

In chapter 4, we consider concentration phenomena of LDPC code ensembles. We show that all except an exponentially (in the block length) small fraction of codes perform within an arbitrary small δ from the ensemble average (where δ is a positive number that can be chosen arbitrarily small). Therefore, assuming a sufficiently long block length, the ensemble average forms a good indicator for the performance of individual codes.

We provide a large-deviation analysis of the conditional entropy of an LPDC code transmitted thorough a MBIOS channel. The derived concentration result tightens a similar bound by Méasson et al. [8, Theorem 4]. For BSC and BEC channels the concentration result is further improved. Compared to the previous bound, the new bound does not only tighten the concentration result, but also expands the concentration result for codes with $d_{\mathrm{c}}^{\max} \to \infty$. For example, a concentration result for a heavy-tail Poisson distribution (Tornado codes) is shown using the refined analysis. The previous result by Méasson et al. can not be used to show concentration in this particular case.

Next, we consider the number of erroneous variable-to-check messages for Inter-Symbol-interference (ISI) channels. The analysis provides an explicit expression for the exponential rate that is related to the concentration inequalities given in [3, Theorems 1 and 2]. It is shown that particularizing these results for memoryless channels provides tightened concentration inequalities as compared to [29] and [48]. Also shown is a bound on the probability of a neighborhood of depth ℓ of a variable-to-check node message to be tree-like for channels with ISI. This result, for channels with ISI, is an extension of the classical result given in [48] for memoryless channels.

5.2 Topics for further research

In what follows, we point out some possible directions for future research:

1. **Bounds on the convergence rate of convex optimization.**
   The newly derived bounds on the number of Newton iterations are 10-100 times tighter than the classical bounds. However, when compared to the numerical results, it is clear that the bounds are still very loose (especially during the damped phase). In the following, we suggest ideas for further tightening of the bounds.

   - Global bound for $\lambda^{(n)}$: The improved bounds use $\lambda^{(n)} > \eta$ in order to bound λ for all the iterations in the damped phase. In practice, the value of λ decreases dramatically during the damped phase. The bound uses a global bound on λ which is tight only at the transition from the damped phase to the quadratic phase. Since the bound on the convergence rate at each iteration during the damped phase is proportional to λ^2, a tight bound for λ at each iteration will dramatically tighten the bound. This is perhaps the main reason why the bounds are not tight at the damped phase.
   - Domain of bounds : During the derivation of the bounds, we used bounds that have a domain of $\lambda t_{\text{line}} < 1$ whereas in practice $\lambda t_{\text{line}} > 1$ in the damped phase. This created an artificial constraint on the line-search's step-size t_{line}. For example, in the first steps of the damped phase $\lambda \gg 1$, this implies that the step-size used in the bound must be very small (i.e.,$t_{\text{line}} < 1/\lambda \ll 1$), whereas in practice a good choice of the step-size is usually close to one. As a result the choice of the predetermined t_{line} is relatively far from optimal and the backtracking bound for t_{bk} is very loose for large values of λ. This in turn causes the bound on the convergence rate $\Delta f(x) > t_{\text{line}} \alpha \lambda^2$ to be very loose. If alternative bounds, without this artificial constraint are found, then much tighter bounds on the convergence rate could be derived.
   - The bounds were derived on the assumption that the objective function is self-concordant. In practice, we are interested in a much narrow family

of functions. Perhaps, if we narrow our discussion only to linear objective functions with additive logarithmic barriers, then a tightened convergence analysis can be performed.

2. **Bounds on the number of iteration required by the IPM-based LP decoder.**

   - Assuming large check-node degree, the fundamental polytope chosen to represent the code requires a large amount of inequalities for its proper description. There exist alternative polytopes that require fewer inequalities. It is interesting to repeat the complexity analysis with a different polytope and compare the complexity bound of alternative representations of the code.
   - As a feasible starting point for the LP decoder, we have chosen the "neutral word" $\mathbf{x}^{(0)} = (1/2, 1/2, ..., 1/2)$. This choice is far from optimal in terms of complexity since it does not use the information from the received word. The algorithm can be optimized if we choose a starting point closer to the ML word. Such a word is the received word, however it is not necessarily feasible. If a variation on the received word which is still feasible is found, then the initial centering step will require less iterations, thus reducing the overall complexity of the IPM-based LP decoder.
   - The IPM is only one method for LP decoding, for example in [22] the simplex method is used for LP decoding. A possible research direction is to perform complexity analysis on the alternative decoders and compare the behavior of the derived bounds.
   - The analysis of the IPM-decoder concentrated on the number of Newton iterations. We note that the time complexity of the IPM-based LP decoder depends not only on the number of Newton iterations but also on the complexity per iteration. At each Newton iteration, we need to calculate the inverse of the hessian matrix (i.e., $\nabla^2 \Psi_t(\mathbf{x}) \in \mathbb{R}^{n \times n}$). It is interesting to analyze the complexity of calculating the inverse of the hessian matrix with respect to the code and channel parameters. Combined with the analysis in this thesis, it will bound the complexity of the IPM-based decoder.

- If in the future, tighter bounds on the number of the Newton method are found, then they can be used in conjunction with the bounds derived here in order to design an IPM-based decoder with search parameters that optimize the Number of iterations of the decoder with respect to the channel and code parameters.
- For LP decoding (with the fundamental polytope given in Definition 11), if there exist a capacity-achieving LDPC ensemble (which is an open question for general MBIOS channel), then we can discuss the case of decoding with vanishing error at rate close to the capacity of the channel. For ML decoding, it is shown in [19] that the maximal right degree (that is, the maximal degree of the parity-check nodes) is not bounded as the gap to capacity vanishes. It grows at least like $\log\left(\frac{1}{\epsilon}\right)$, where ϵ is the fractional gap to capacity. If a similar behavior can be shown for the considered LP decoder then the bound on the number of iterations scales like $O\left(\frac{1}{\epsilon^k}\right)$, where $k > 0$ is some positive constant (since the bound scales like $O(2_{\mathrm{c}}^d)$). This bound is similar to the conjecture regarding iterative message-passing decoding that claims that the number of iterations scales like $O\left(\frac{1}{\epsilon}\right)$ (with a proof for the BEC given in [17]).

3. **Concentration of measures of LDPC code ensembles.**

- The concentration result given in Theorem 17 is very loose. This is because the proof is very pessimistic. We assume that any change in an edge or the decoder's input will introduce an error in every message it affects. This is especially pessimistic if large neighborhoods are considered. In practice, the probability that changing a single edge or input will change the message is close to zero for long codes and large neighborhoods. A refined analysis should take into account the probability for mistakes as a function of the neighborhood size and/or the structure of the parity-check matrix/tanner graph of the code. Furthermore, if we apply the concentration result to irregular codes, we need to use $d_{\mathrm{c}}^{\max}$, $d_{\mathrm{v}}^{\max}$ as a bound for d_{c} and d_{v} respectively. A refined analysis should take into account the degree distribution of the irregular code.
- The use of the union bound in the concentration result of M. Sipser and

D. A. Spielman for the graph expansion [34] seems to weaken this theorem significantly. Perhaps, it is possible to improve the bound in this theorem by replacing the use of the union bound with a refined analysis.

- The utilization of refined inequalities of the Azuma-Hoeffding inequality in the context of graph-based codes.

Appendix A

Proof of Lemma 2

Proof : Let $v = -\nabla^2 f(x)^{-1} \nabla f(x)$, then from [45, Ex. (9.17)]

$$(1 - t\lambda(x))^2 \nabla^2 f(x) \preceq \nabla^2 f(x + tv) \preceq \frac{\nabla^2 f(x)}{(1 - t\lambda(x))^2}.$$

We first prove (2.32) for the case where $\nabla^2 f(x) = I$. Using $v = -\nabla f(x)$ we get a simplified bound for $\nabla^2 f(x + tv)$

$$(1 - t\lambda(x))^2 I \preceq \nabla^2 f(x + tv) \preceq \frac{1}{(1 - t\lambda(x))^2} I. \tag{A.1}$$

The function $\nabla f\left(x^+\right)$ can be expressed using $\nabla f\left(x\right)$ as follows

$$\begin{aligned} \nabla f\left(x^+\right) &= \nabla f\left(x + tv\right) \\ &= \int_0^t \nabla^2 f\left(x + t'v\right) v dt' + \nabla f(x) \\ &= \left[I - \int_0^t \nabla^2 f(x + t'v) dt' \right] \nabla f(x) \\ &= A \cdot \nabla f\left(x\right). \end{aligned} \tag{A.2}$$

where $A \equiv I - \int_0^t \nabla^2 f\left(x + t'v\right) dt'$. Applying (A.1) on the expression for A yields the following bound

$$aI \preceq A \preceq bI \tag{A.3}$$

where

$$\begin{aligned} a &= 1 - \int_0^t \frac{1}{\left(1-\lambda t'\right)^2} dt' = 1 - \frac{t}{1-\lambda t} \\ b &= 1 - \int_0^t \left(1-\lambda t'\right)^2 dt' = 1 - \frac{1-(1-\lambda t)^3}{3\lambda}. \end{aligned}$$

Moreover, since the hessian is a symmetric matrix, then A is a symmetric matrix as well. Using $A^T A = A^2$ and (A.3) we get a bound on $A^T A$

$$\left(\max\left(0,a\right)\right)^2 \cdot I \preceq A^2 = A^T A \preceq \left(\max\left(|a|,|b|\right)\right)^2 \cdot I \tag{A.4}$$

Using the properties above we can write $\lambda\left(x^+\right)$ as follows

$$\begin{aligned} \lambda\left(x^+\right) &\overset{(a)}{=} \sqrt{\nabla f\left(x^+\right)^T \nabla^2 f\left(x^+\right)^{-1} \nabla f\left(x^+\right)} \\ &\overset{(b)}{\leq} \frac{1}{1-t\lambda(x)} \sqrt{\nabla f\left(x^+\right)^T \nabla f\left(x^+\right)} \\ &\overset{(c)}{=} \frac{1}{1-t\lambda(x)} \sqrt{\nabla f(x)^T \left(A^T A\right) \nabla f(x)} \\ &\overset{(d)}{\leq} \left(\frac{\max(|a|,|b|)}{1-t\lambda(x)}\right) \left\|\nabla f(x)\right\|_2 \\ &\overset{(e)}{=} c(t,\lambda(x))\lambda(x). \end{aligned}$$

Where (a) follows from the definition of λ, (b) follows from $\left(\nabla^2 f\left(x^+\right)\right)^{-1} \preceq \frac{1}{(1-t\lambda(x))^2}$ which results from (A.1) by inversion of the lower bound, (c) follows from the expression for $\nabla f\left(x^+\right)$ in terms of $\nabla f\left(x\right)$ given in (A.3), (d) follows from the upper bound for $A^T A$ in (A.4) and (e) follows from the definition of $c(t,\lambda)$ in (2.33) and since $\lambda(x) = \left\|\nabla f(x)\right\|_2$ for $\nabla^2 f(x) = I$.

We now generalize this result to a strictly convex s.c. function with an arbitrary $\nabla^2 f(x) \succ 0$ (i.e.,$\nabla^2 f(x) \neq I$). This is done by showing that there exist a linear transformation $\tilde{x} = Tx$ on $\tilde{f}$ (where $\nabla^2 \tilde{f}\left(\tilde{x}\right) = I$) which will result in $\nabla^2 f(x) = T^T \nabla^2 \tilde{f}\left(\tilde{x}\right) T$. Since the Newton decrement λ is invariant to linear transformation the derivation made for $\tilde{f}\left(\tilde{x}\right)$ will still hold for $f(x)$.

The function f is strictly convex, thus $\nabla^2 f(x)$ is symmetric with positive eigenvalues. This means it can be factored in the form $\nabla^2 f(x) = Q^T \Lambda Q$, where $Q^T = Q^{-1}$

(Orthonormal matrix) and $\Lambda = \text{diag}\left(\{\lambda_i > 0\}\right)$. We can further expand $\nabla^2 f(x)$ as follows $\nabla^2 f(x) = Q^T \Lambda Q = \left(\Lambda^{1/2} Q\right)^T I \left(\Lambda^{1/2} Q\right) \underset{T \equiv \Lambda^{1/2} Q}{=} T^T \nabla^2 \tilde{f}\left(\tilde{x}\right) T$. Thus we can use $T = \Lambda^{1/2} Q$ as the transformation required in the previous paragraph.

Appendix B

Proof of Lemma 5

In order to prove Lemma 5, one needs to show that if $\rho'(1) < \infty$ then

$$\lim_{C \to 1} \sum_{i=1}^{\infty} (i+1)^2 \Gamma_i \left[h_2 \left(\frac{1 - C^{\frac{i}{2}}}{2} \right) \right]^2 = 0 \tag{B.1}$$

which then yields from (4.11) that $B \to \infty$ in the limit where $C \to 1$.

By the assumption in Lemma 5 where $\rho'(1) < \infty$ then $\sum_{i=1}^{\infty} i\rho_i < \infty$, and therefore it follows from the Cauchy-Schwarz inequality that

$$\sum_{i=1}^{\infty} \frac{\rho_i}{i} \geq \frac{1}{\sum_{i=1}^{\infty} i\rho_i} > 0.$$

Hence, the *average* degree of the parity-check nodes is finite

$$d_{\mathrm{c}}^{\mathrm{avg}} = \frac{1}{\sum_{i=1}^{\infty} \frac{\rho_i}{i}} < \infty.$$

The infinite sum $\sum_{i=1}^{\infty} (i+1)^2 \Gamma_i$ converges under the above assumption since

$$\begin{aligned} &\sum_{i=1}^{\infty} (i+1)^2 \Gamma_i \\ &= \sum_{i=1}^{\infty} i^2 \Gamma_i + 2 \sum_{i=1}^{\infty} i\Gamma_i + \sum_{i} \Gamma_i \\ &= d_{\mathrm{c}}^{\mathrm{avg}} \left(\sum_{i=1}^{\infty} i\rho_i + 2 \right) + 1 < \infty. \end{aligned}$$

where the last equality holds since

$$\begin{aligned}\Gamma_i &= \frac{\frac{\rho_i}{i}}{\int_0^1 \rho(x)\,\mathrm{d}x} \\ &= d_{\mathrm{c}}^{\mathrm{avg}}\left(\frac{\rho_i}{i}\right), \quad \forall\, i \in \mathbb{N}.\end{aligned}$$

The infinite series in (B.1) therefore uniformly converges for $C \in [0, 1]$, hence, the order of the limit and the infinite sum can be exchanged. Every term of the infinite series in (B.1) converges to zero in the limit where $C \to 1$, hence the limit in (B.1) is zero. This completes the proof of Lemma 5.

References

[1] A. Barg and G. D. Forney, "Random codes: minimum distances and error exponents," *IEEE Trans. on Information Theory*, vol. 48, no. 9, pp. 2568–2573, September 2002.

[2] A. G. Dimakis and M. J.Wainwright, "Guessing facets: Polytope structure and improved LP decoder," in Proc. *IEEE Int. Symp. Information Theory*, Seattle, WA, Jul. 2006.

[3] A. Kavcić, X. Ma and M. Mitzenmacher, "Binary intersymbol interference channels: Gallager bounds, density evolution, and code performance bounds,"*IEEE Trans. on Information Theory*, vol. 49, no. 7, pp. 1636—1652, July 2003.

[4] A. Montanari, "Tight bounds for LDPC and LDGM codes under MAP decoding," *IEEE Trans. on Information Theory*, vol. 51, no. 9, pp. 3247–3261, September 2005.

[5] A. Nemirovsky, *Interior Point Polynomial Time Methods in Convex Programming*, Lecture Notes,2004. [Online]. Available: `www2.isye.gatech.edu/~nemirovs/Lect_IPM.pdf`.

[6] A. Shokrollahi, "Capacity-achieving sequences," *IMA Volume in Mathematics and its Applications*, vol. 123, pp. 153–166, 2000.

[7] C. McDiarmid, "Concentration," *Probabilistic Methods for Algorithmic discrete mathematics*, pp. 1–46, 1998.

[8] C. Méasson, A. Montanari and R. Urbanke, "Maxwell construction: The hidden bridge between iterative and maximum apposteriori decoding," *IEEE Trans. on Information Theory*, vol. 54, pp. 5277–5307, December 2008.

[9] C. Roos, T. Terlaky, and J. Vial, *Interior Point Methods for Linear Optimization*, 2nd ed. New York, springer,2006.

[10] D. Burshtein, "Iterative approximate linear programming decoding of LDPC codes with linear complexity," in Proc. *IEEE Int. Symp. Information Theory*, Toronto, Canada, Jul. 2008.

[11] D. Burshtein, "Iterative Approximate Linear Programming Decoding of LDPC Codes with Linear Complexity," *IEEE Transactions on Information Theory*, vol. 55, no. 11, pp. 4835-4859, November 2009

[12] D. Mackay, "Good error corecting codes based on vey sparse matrices", *IEEE Trans. on Information Theory*, vol. 45, no. 2, pp. 399–431, 1991.

[13] D. P. Dubashi and A. Panconesi, *Concentration of Measure for the Analysis of Randomized Algorithms*, Cambridge University Press, 2009.

[14] F. Chung and L. Lu, "Concentration inequalities and martingale inequalities: a survey," *Internet Mathematics*, vol. 3, no. 1, pp. 79–127, March 2006.

[15] I. Sason, "On universal properties of capacity-approaching LDPC code ensembles," *IEEE Trans. on Information Theory*, vol. 55, no. 7, pp. 2956–2990, July 2009.

[16] I. Sason, "On the fundamental system of cycles in the bipartite graphs of LDPC code ensembles," *Proceedings 2009 IEEE International Symposium on Information Theory (ISIT 2009)*, pp. 75–79, Seoul, Korea, June 28–July 3, 2009.

[17] I. Sason and G. Wiechman, "Bounds on the number of iterations for turbo-like ensembles over the binary erasure channel," *IEEE Trans. on Information Theory*, vol. 55, no. 6, pp. 2602 - 2617, June 2009.

[18] I. Sason and R. Eshel "On concentration of measures for LDPC code ensembles," *Proceedings 2011 IEEE International Symposium on Information Theory (ISIT 2011)*, pp. 1268–1272, St. Petersburg, Russia, July 31–Aug 5, 2011

[19] I. Sason and R. Urbanke, "Parity-check density versus performance of binary linear block codes over memoryless symmetric channels," *IEEE Trans. on Information Theory*, vol. 49, no. 7, pp. 1611 - 1635, July 2003.

[20] J. Douillard, M. Jezequel, C. Berrou, A. Picart, P. Didier, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," Europ. Trans. Commun., vol. 6, pp. 507—511, Sept. 1995.

[21] J. Feldman, "decoding Error-Correcting Codes via Linear Programming", Ph.D, Mass. Inst. Technology, Cambridge, 2003.

[22] J. Feldman, M.J. Wainwright, and D.R. Karger, "Using linear programming to decode binary linear codes", *IEEE Trans. on Information Theory*, vol. 51, no. 3, pp. 954–972, Mar. 2005.

[23] J. S. Rosenthal, *A First look at Rigorous Probability Theory*, World Scientific Publishes, second edition, 2006.

[24] K. Azuma, "Weighted sums of certain dependent random variables," *Tohoku Mathematical Journal*, vol. 19, pp. 357–367, 1967.

[25] K. Yang, X. Wang, and J. Feldman, "A new linear programming approach to decoding linear block codes," *IEEE Trans. Inf. Theory*, vol. 54, no. 3, pp. 1061–1072, Mar. 2008.

[26] M. Breiling, "A logarithmic upper bound on the minimum distance of turbo codes," *IEEE Trans. on Information Theory*, vol. 50, pp. 1692–1710, August 2004.

[27] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Improved low density parity check codes using irregular graphs and belief propagation" *Proceedings 1998 IEEE International Symposium on Information Theory*, Cambridge, MA, Aug. 1998, p. 117.

[28] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi and D. A. Spielman, "Efficient erasure-correcting codes," *IEEE Trans. on Information Theory*, vol. 47, no. 2, pp. 569–584, February 2001.

[29] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi and D. A. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. on Information Theory*, vol. 47, no. 2, pp. 585–598, February 2001.

[30] M. H. Taghavi and P. H. Siegel, "Adaptive linear programming decoding," in *Proc. IEEE Int. Symp. Information Theory*, Seattle, WA, Jul. 2006.

[31] M. H. Taghavi and P. H. Siegel, "Adaptive methods for linear programming decoding," *IEEE Trans. Inf. Theory*, vol. 54, no. 12, pp. 5396–5410, Dec. 2008.

[32] M. Ledoux, *The Concentration of Measure Phenomenon*, Mathematical Surveys and Monographs (AMS), vol. 89, 2001.

[33] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*, Cambridge University Press, Cambridge, MA, USA, 2005.

[34] M. Sipser and D. Spielman, "Expander codes", *IEEE Trans. on Information Theory*, vol. 42, no. 6, pp. 1710–1722, 1996.

[35] M. Talagrand, "Concentration of measure and isoperimteric inequalities in product spaces," *Publications Mathématiques de l'I.H.E.S*, vol. 81, pp. 93–205, 1995.

[36] N. Alon and J. H. Spencer, *The Probabilistic Method*, Wiley Series in Discrete Mathematics and Optimization, Third Edition, 2008.

[37] N. K. Karmarkar, *A new polynomial-time algorithm for linear programming* , Combinatorica 4 (1984), 373–395.

[38] N. Wiberg, "Codes and Decoding on General Graphs", PhD thesis, Linkoping University, Sweden, 1996.

[39] P. Oswald and A. Shokrollahi, "Capacity-achieving sequences for the erasure channel," *IEEE Trans. on Information Theory*, vol. 48, no. 12, pp. 3017–3028, December 2002.

[40] P. O. Vontobel, "Interior-Point Algorithms for Linear-Programming Decoding," in *Information Theory and Applications Workshop* , 2008, pp. 433–437, DOI 10.1109/ITA.2008.4601085

[41] P. O. Vontobel and R. Koetter, "Towards low-complexity linear-programming decoding," in Proc. 4th Int. Symp. Turbo Codes and Related Topics, Munich, Germany, Apr. 2006.

[42] R. Gallager. "Low-desity parity-check codes" *IRE Trans. Information Theory*,Vol 8, pp. 21-28, Jan, 1962.

[43] R. McEliece, D. Mackay and J. Cheng, "Turbo decoding as an instance of pearl's belief propagation algorithm", *IEEE Journal on Selected Areas in Communictions*, vol. 16, no. 2, pp. 140–152, 1998.

[44] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, Cambridge, MA, USA, 1995.

[45] S. Boyd and L. Vanderberghe, *Convex Optimization*, Cambridge Press, 2004. [Online]. Available: `http://www.stanford.edu/~boyd/cvxbook/`.

[46] S. Y. Chung, G. D. Forney, T. Richardson, and R. Urbanke, " On the design of low-desity parity check codes within 0.0045dB of the Shannon limit", *IEEE Communications Letters*, vol. 5, no. 2, pp. 58–60, February 2001.

[47] T. Etzion, A. Trachtenberg and A. Vardy, "Which codes have cycle-free Tanner graphs?," *IEEE Trans. on Information Theory*, vol. 45, no. 6, pp. 2173–2181, September 1999.

[48] T. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message-passing decoding," *IEEE Trans. on Information Theory*, vol. 47, pp. 599–618, February 2001.

[49] T. Richardson and R. Urbanke, *Modern Coding Theory*, Cambridge University Press, 2008.

[50] T. Wadayama, "An LP decoding algorithm based on primal path-following interior point method," *Proceedings 2009 IEEE International Symposium on Information Theory*, pp. 389–394, Seoul, South Korea, June 28 - July 3, 2009.

[51] T. Wadayama, "Interior point decoding for linear vector channels based on convex optimization," *IEEE Trans. on Information Theory*, vol. 56, no. 10, pp. 4905–4921, October 2010.

[52] Y. E. Nesterov and A. S. Nemirovski, *Interior-point polynomial methods in convex programming*, SIAM Studies in Applied Mathematics, SIAM Publications, Philadelphia, 1994.

density evolution. על מנת להוכיח את תופעות ההתרכזות למקרים מסויימים ולקבל תוצאות הדוקות יותר ממספר תוצאות קלאסיות, אנו משתמשים בתכונות של מרטינגלים בעלי הפרשים חסומים. על ידי שימוש באי שיוויון Azuma Hoeffding אנו מבצעים אנליזה משופרת ביחס למספר תוצאות שפורסמו בעבר. התוצאה הראשונה הינה תוצאת התרכזות מהודקת בנוגע לאנטרופיה המותנת של מילת הקוד בהינתן הקלט מהערוץ כאשר התשדורת מתרחשת בערוץ בינארי, סימטרי וחסר זיכרון (MBIOS). כמו כן עבור ערוצי BSC ו BEC אנו מראים כי ניתן להגיע להידוק נוסף של החסם. בנוסף לכך, החסם מאפשר להראות תוצאות התרכזות עבור מספר מקרים מעניינים שבהם החסמים הקודמים היו חסרי משמעות (לדוגמא עבור קודי Tornado שמתאפיינים בהתפלגות פואסונית של דרגות בדיקת הזוגיות). בהמשך אנו מספקים ביטויים מפורשים למספר תוצאות התרכזות המתייחסות לתשדורת בערוצים עם הפרעות בין סימניות (ISI) בעלי זיכרון בעומק I ופענוח ע"י מפענח מסוג windowed min sum בעל חלון ברוחב W. התוצאה הראשונה הינה חסם להסתברות שסביבה מדרגה ℓ של הודעה בין צומת משתנה לצומת בדיקת זוגיות איננה עץ. התוצאה הינה הרחבה של תוצאות קודמות לערוצים חסרי זיכרון והיא אכן מתלכדת עם תוצאות אילו אם מניחים שאין הפרעות בין סימבולים. בהמשך אנו מתייחסים להתרכזות של מספר ההודעות השגויות בין צומת משתנה לצומת בדיקת זוגיות. האנליזה מספקת ביטויים מפורשים לגבי הקצב המעריכי הקשור לאי שוויוני ההתרכזות. לבסוף אנו מראים כי ניוון התוצאות לערוצים חסרי זיכרון סימטריים מהדק את תוצאת ההתרכזות שהוכחה על ידי Luby et al. ו Urbanke & Richardson. על אף השיפורים בהדיקות אי שיוויוני ההתרכזות, התוצאות הינן עדיין פסימיות. סיבת אמת לכך היא שאנו מניחים שכל שינוי בקוד או בערוץ עלול להכניס שגיאות בעוד שבפועל הסיכוי למאורע כזה נמוך.

לסיום, אנו סוקרים כיוונים למחקרים נוספים בתחום, ומספקים דוגמאות לתחומים שדורשים התייחסות נוספת.

על ידי אלגוריתמים איטרטיביים מסוג belief propafation (לדוגמא מפענחי min sum ו product sum) שבהם ההודעות נשלחות בצורה איטרטיבית על פני קשתות הגרף שמייצג את הקוד. בשנים האחרונות מפענחים חילופיים המתבססים על תכנות לינארי החלו לצבור עניין אקדמי. מפענח תכנות ליניארי מתבסס על העובדה כי תחת הנחות מסוימות ניתן לייצג את מפענח ה likelihood Maximum כבעיית אופטימיזציה בעלת פונקצית מטרה ליניארית ומרחב פתרון בדיד. על מנת להפוך את הבעיה לבעיית LP שהיינה פתירה מבחינה חישובית, אנו מבצעים הרפיה (relaxation) של מרחב הפתרונות (כגון מעבר למרחב רציף). לאחר הצגת מפענח תכנות ליניארי המבוסס על שיטת IPM אנו מיישמים את החסמים מהחלק הקודם על מפענח LP על מנת לקבל חסם על מספר איטרציות ניוטון שמבצע המפענח. בהמשך, אנו מבצעים אופטימיזציה על הפרמטרים של המפענח הקשורים לשיטת ה IPM על מנת להביא להידוק נוסף של החסם. פרמטרים אילו עתה תלויים בפרמטרים של הערוץ והקוד, כגון : אורך הבלוק, דרגת מטריצת בדיקת הזוגיות, רמת הרעש וכד'. החסמים החדשים מאפשרים תובנות אנליטיות על מספר איטרציות ניוטון שמבצע המפענח כתלות בפרמטרי הקוד והערוץ. החסם מראה כי מספר האיטרציות כתלות באורך הבלוק הינו מעריכי באופן כללי, אולם עבור קודי LDPC ניתן להראות כי מספר האיטרציות חסום על ידי $O(\sqrt{n}\ln(n))$. באופן דומה הסיבוכיות של המפענח (שלוקחת בחשבון את הסיבוכיות הכרוכה בכל איטרציה של הפענוח) חסומה על ידי $O(n^{1.5}\ln(n))$. לפיכך מקודד זה מתאים לפענוח קודי LDPC. בהמשך, אנו מראים כי מספר האיטרציות עולה באופן מעריכי כתלות בדרגת מטריצת בדיקת הזוגיות של הקוד. לבסוף אנו מראים כי מספר האיטרציות חסום על ידי שורש מספר אילוצי האי שיוויון הדרושים לתיאור הקוד. מכיוון שביצועי המפענח עולים ככל שמספר האילוצים גדל, קיים trade off בין סיבוכיות המפענח לביצועיו. לבסוף אנו בוחנים את של החסם באופן נומרי עבור קוד ושיטת פענוח ספציפיים. החסם המשופר מהווה שיפור ביחס לחסמים דומים שניתנו על ידי Wadayama, אולם הוא עדיין איננו הדוק. ע"י שימוש בחסם שעבר אופטימיזציה בשילוב עם השיטות החדשות לחישוב גודל הצעד מתקבל שיפור משמעותי בהדיקות החסם. במידה והמקדמים הקשורים לחסם על מספר האיטרציות באופטימיזציה ללא אילוצים יהודקו בעתיד, ניתן יהיה להשתמש בהם בשילוב עם החסם שהוצג בחלק זה על מנת לקבל אומדן טוב יותר לגבי מספר האיטרציות שמבצע המפענח וכן לאפשר בחירה אופטימאלית של פרמטרי המפענח כתלות בפרמטרי הערוץ והקוד.

בחלקו האחרון של העבודה, אנו בוחנים תופעות התרכזות מידות (concentration of measures) בצבירים של קודי LDPC. תוצאות אילו, בהמשך לתוצאות שפורסמו בעבר במהלך 15 השנים האחרונות, מראות כי עבור צביר קודים נתון, כמעט כל הקודים פרט לחלק זעיר מהצביר מתנהג כמו הממוצע של על פני צביר הקודים. לפיכך עבור אורכי בלוק גדולים, ההתנהגות הממוצעת של צביר הקודים מהווה אינדיקציה טובה להתנהגות קודים ספציפיים מהצביר ומהווה את הצידוק האנליטי לניתוח הסטטיסטי של הביצועים האסימפטוטים של צביר קודי LDPC בעזרת שיטת

תקציר

עבודה זו מתייחסת לאספקטים של אופטימיזציה קמורה והתרכזות מידות בהיבט של בעיות קידוד בתקשורת. ניתן לחלק את התזה לשלושה חלקים עיקריים.

החלק הראשון מתייחס לפיתוח חסמים עליונים הקשורים לסיבוכיות של פתרון בעיות אופטימיזציה קמורות עם פונקציית מטרה מסוג self concordant (s.c.). על ידי שימוש בתכונות של פונקציות מסוג s.c., אנו מוכיחים חסמים עליונים על מספר האיטרציות הנדרשות על ידי אלגוריתם ניוטון לפתרון של בעיית אופטימיזציה קמורה בעלת אילוצי שיוויון. החסם הראשון ניתן עבור שיטת backtracking לחישוב גודל צעד ניוטון. החסם העליון הנגזר בעבודה זו משווה לחסם קלאסי, ומראה גם הוא שההתכנסות מחולקת לשתי פאזות: damped phase שבמהלכה קיימת התכנסות מתונה של האלגוריתם, ולאחר מכן quadratic phase שבמהלכה קיימת התכנסות מהירה מאוד של האלגוריתם. השוואת חסם זה לחסמים דומים מהעבר מראה על שיפור של בערך פי 10 עד 100 בהדיקות החסם. כמו כן מתקבל כי התנהגות החסם כתלות בפרמטרי שיטת ה backtracking דומה להתנהגות של הסיבוכיות בפועל. באופן דומה אנו מוכיחים חסמים עבור שתי צורות נוספות של חישוב גודל הצעד. השיטות מבוססות על חישוב של גודל הצעד על ידי פונקצייה ידוע מראש והן נבחרו כך שהן מהדקות את החסם על מספר האיטרציות באופן שמאפשר להגיע לשיפור נוסף בהדיקות החסם. בהתבסס על פיתוחים דומים מהעבר, אנו מרחיבים חסמים אילו עבור בעיות עם אילוצי אי שוויון ושימוש בשיטת interior point method (IPM). בעזרת סימולציות ממוחשבות אנו מבצעים אנליזה נומרית שבוחנת את טיבם של החסמים המהודקים שהוכחנו. הסימולציות מראות כי החסמים מנבאים את ההתנהגות הכללית של סיבוכיות הבעיה אולם אינם מאוד הדוקים. בעוד שב quadratic phase החסמים הדוקים באופן יחסי, הרי שב damped phase קיים מקום לשיפור נוסף של החסם. בעזרת דוגמאות נומריות אנו מנתחים את הסיבות לחוסר הדיקות זו, ומראים כי על ידי שיפור זוג גורמים בפיתוח ניתן להגיע לחסם הדוק באופן יחסי. כמו כן, כחלק מתהליך פיתוח החסמים אנו מרחיבים תוצאה הנוגעת לחסם רקורסיבי על ה Newton decrement למקרה שבו גודל צעד ניוטון איננו יחידה.

החלק השני של התזה מתייחס לבעיה של פענוח קודי בלוק ליניאריים ובינאריים (לדוגמא, קודי LDPC) בעזרת מפענח המבוסס על תכנות ליניארי (LP). לרוב, פענוח קודי LDPC מבוצע

להורי רות וראובן אשל.

המחקר נעשה בהדרכת פרופ' יגאל ששון
בפקולטה להנדסת חשמל

הכרת תודה

תודות חמות מגיעות למנחה שלי, פרופ' יגאל ששון, על הנחייתו הצמודה והמסורה לכל אורך הדרך. ללא עזרתו, לא היתה עבודה זו מגיעה לכדי השלמה.
לבסוף, אני מודה להורי, רות וראובן אשל, על תמיכתם והסבלנות שגילו לכל אורך תקופת השתלמותי. עבודה זו מוקדשת לכם.

אני מודה לטכניון על התמיכה הכספית הנדיבה בהשתלמותי

אספקטים של אופטימיזציה קמורה והתרכזות מידות בבעיות קידוד בתקשורת

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת תואר

מגיסטר למדעים

הנדסת חשמל

רונן אשל

הוגש לסנט הטכניון — מכון טכנולוגי לישראל

שבט תשע"ב חיפה פברואר 2012

אספקטים של אופטימיזציה קמורה והתרכזות מידות בבעיות קידוד בתקשורת

רונן אשל