Sparse Regression Codes

Ramji Venkataramanan, *University of Cambridge*

Sekhar Tatikonda, *Yale University*

Andrew Barron, *Yale University*

Contents

Abstract

Developing computationally-efficient codes that approach the Shannon-theoretic limits for communication and compression has long been one of the major goals of information and coding theory. There have been significant advances towards this goal in the last couple of decades, with the emergence of turbo codes, sparse-graph codes, and polar codes. These codes are designed primarily for discrete-alphabet channels and sources. For Gaussian channels and sources, where the alphabet is inherently continuous, *Sparse Superposition Codes* or *Sparse Regression Codes* (SPARCs) are a promising class of codes for achieving the Shannon limits.

This monograph provides a unified and comprehensive over-view of sparse regression codes, covering theory, algorithms, and practical implementation aspects. The first part of the monograph focuses on SPARCs for AWGN channel coding, and the second part on SPARCs for lossy compression (with squared error distortion criterion). In the third part, SPARCs are used to construct codes for Gaussian multi-terminal channel and source coding models such as broadcast channels, multiple-access channels, and source and channel coding with side information. The monograph concludes with a discussion of open problems and directions for future work.

Chapter 1

Introduction

Developing computationally-efficient codes that approach the Shannon-theoretic limits for communication and compression has long been one of the major goals of information and coding theory. There have been significant advances towards this goal in the last couple of decades, with the emergence of turbo and sparse-graph codes in the '90s [21, 30, 93], and more recently polar codes and spatially-coupled LDPC codes [5, 69, 74]. These codes are primarily designed for channels with discrete input alphabet, and for discrete-alphabet sources.

There are many channels and sources of practical interest where the alphabet is inherently continuous, e.g., additive white Gaussian noise (AWGN) channels, and Gaussian sources. This monograph discusses a class of codes for such Gaussian models called *Sparse Superposition Codes* or *Sparse Regression Codes* (SPARCs). These codes were introduced by Barron and Joseph [16, 65] for efficient communication over AWGN channels, but have since also been used for lossy compression [112, 113] and multi-terminal communication [114]. Our goal in this monograph is to provide a unified and comprehensive view of SPARCs, covering theory, algorithms, as well as practical implementation aspects.

To motivate the construction of SPARCs, let us begin with the standard AWGN channel. The goal is to construct codes with computationally efficient encoding and decoding that *provably* achieve the channel capacity $\mathcal{C} = \frac{1}{2} \log_2(1 + \mathsf{snr})$ bits/transmission, where snr denotes the signal-to-noise ratio. In particular, we are interested in codes whose encoding and decoding complexity grows no faster than a low-order polynomial in the block length n.

Though it is well known that rates approaching $\mathcal{C}$ can be achieved with Gaussian codebooks, this has been largely avoided in practice because of the high decoding complexity of unstructured Gaussian codes. Instead, the popular approach has been to separate the design of the coding scheme into two steps: *coding* and *modulation*. State-of-the-art coding schemes for the AWGN channel such as coded modulation [45, 52, 23] use this two-step design, and combine binary error-correcting codes such as LDPC and turbo codes with standard modulation schemes such as Quadrature Amplitude Modulation (QAM). Though such schemes have good empirical performance, they have not been proven to be capacity-achieving for the AWGN channel. With sparse regression codes, we step back from the coding/modulation divide and instead use a structured codebook to construct low-

Figure 1.1: A Gaussian sparse regression codebook of block length n: A is a design matrix with independent Gaussian entries, and β is a sparse vector with one non-zero in each of L sections. Codewords are of the form $A\beta$, i.e., linear combinations of the columns corresponding to the non-zeros in β. The message is indexed by the *locations* of the non-zeros, and the values $c_1, \ldots, c_L$ are fixed a priori.

complexity, capacity-achieving schemes tailored to the AWGN channel.

There have been several lattice based schemes [42, 121] proposed for communication over the AWGN channel, including low density lattice codes [102] and polar lattices [118, 3]. The reader is referred to the cited works for details of the performance vs. complexity trade-offs of these codes.

In the rest of this chapter, we describe the sparse regression codebook, and give a brief overview of the topics covered in the later chapters. First, we lay down some notation that will be used throughout the monograph.

Notation The Gaussian distribution with mean μ and variance σ^2 is denoted by $\mathcal{N}(\mu, \sigma^2)$. For a positive integer L, we use $[L]$ to denote the set $\{1, \ldots, L\}$. The Euclidean norm of a vector x is denoted by $\|x\|$. The indicator function of an event $\mathcal{E}$ is denoted by $\mathbf{1}\{\mathcal{E}\}$. The transpose of a matrix A is denoted by A^*. The $n \times n$ identity matrix is denoted by I_n, with the subscript dropped when it is clear from context.

Both log and ln are used to denote the natural logarithm. Logarithms to the base 2 are denoted by $\log_2$. For most of the theoretical analysis, we will find it convenient to use natural logarithms. Therefore, rate is measured in *nats*, unless otherwise specified. Throughout, we use n for the block length of the code.

For random vectors X, Y defined on the same probability space, we write $X \overset{d}{=} Y$ to indicate that X and Y have the same distribution.

1.1 The Sparse Regression Codebook

As shown in Fig. 1.1, a SPARC is defined in terms of a 'dictionary' or design matrix A of dimension $n \times ML$, whose entries are chosen i.i.d. $\sim \mathcal{N}(0, \frac{1}{n})$. Here n is the block length, and M, L are integers

whose values are specified below in terms of n and the rate R. We think of the matrix A as being composed of L sections with M columns each. The variance of the entries ensures that the lengths of the columns of A are close to 1 for large n. [1]

Each codeword is a linear combination of L columns, with exactly one column chosen per section. Formally, a codeword can be expressed as $A\beta$, where $\beta = (\beta_1, \ldots, \beta_{ML})^*$ is a length ML message vector with the following property: there is exactly one non-zero β_j for $1 \leq j \leq M$, one non-zero β_j for $M + 1 \leq j \leq 2M$, and so forth. We denote the set of valid message vectors by $\mathcal{B}_{M,L}$. Since each of the L sections contains M columns, the size of this set is

$$|\mathcal{B}_{M,L}| = M^L. \tag{1.1}$$

The non-zero value of β in section $\ell \in [L]$ is set to c_ℓ, where the coefficients $\{c_\ell\}$ are specified a priori. Since the entries of A are i.i.d. $\mathcal{N}(0, \frac{1}{n})$, the entries of the codeword $A\beta$ are i.i.d. $\mathcal{N}(0, \frac{1}{n} \sum_{\ell=1}^{L} c_\ell^2)$. In the case of AWGN channel coding, the variance $\frac{1}{n} \sum_{\ell=1}^{L} c_\ell^2$ is equal to the average symbol power.

Rate: Since each of the L sections contains M columns, the total number of codewords is M^L. To obtain a rate R code, we need

$$M^L = e^{nR} \quad \text{or} \quad L \log M = nR. \tag{1.2}$$

There are several choices for the pair (M, L) which satisfy (1.2). For example, $L = 1$ and $M = e^{nR}$ recovers the Shannon-style random codebook in which the number of columns in A is e^{nR}. For most of our constructions, we will often choose M equal to L^{a}, for some constant $\mathsf{a} > 0$. In this case, (1.2) becomes

$$\mathsf{a} L \log L = nR. \tag{1.3}$$

Thus $L = \Theta(\frac{n}{\log n})$, and the size of the design matrix A (given by $n \times ML = n \times L^{\mathsf{a}+1}$) grows polynomially in n. In our numerical simulations, typical values for L are 512 or 1024.

We note that the SPARC is a non-linear code with pairwise dependent codewords. Indeed, two codewords $A\beta$ and $A\beta'$ are dependent whenever the underlying message vectors β, β' share one or more common non-zero entries.

Subset superposition coding   The SPARC described above has a partitioned structure, i.e., the message vector contains exactly one non-zero in each of the L sections, with each section having M entries. One could also define a non-partitioned SPARC, where a message can be indexed by *any* subset of L entries of the length-ML vector β. The number of codewords in this case would be $\binom{ML}{L}$, compared to M^L for the partitioned case. For a given pair (M, L), the non-partitioned SPARC has a larger number of codewords. However, using Stirling's formula we find that

$$\frac{\log \binom{ML}{L}}{\log M^L} = 1 + O\left(\frac{1}{\log M}\right).$$

Hence the ratio of the rates tends to 1 as M grows large. Though subset based (non-partitioned) superposition codes have a small rate advantage for finite M, we focus on the partitioned structure in this monograph as it facilitates the design and analysis of efficient coding algorithms.

---

[1]In some papers, the entries of A are assumed to be $\sim_{i.i.d.} \mathcal{N}(0,1)$. For consistency, throughout this monograph we will assume that the entries are $\sim_{i.i.d.} \mathcal{N}(0, 1/n)$.

Figure 1.2: Average bit error rate (left) and codeword error rate (right) vs. rate for SPARC over an AWGN channel with snr $= 15$, $\mathcal{C} = 2$ bits. The SPARC parameters are $M = 512$, $L = 1024$, $n \in [5100, 7700]$. Curves are shown for for power allocated SPARC (Chapter 4) and spatially coupled SPARC (Chapter 5). The different ways of measuring error rate performance in a SPARC are discussed in Chapter 2 (p.10). The SPARC is decoded using the Approximate Message Passing (AMP) algorithm described in Chapters 3 and 5.

1.2 Organization of the monograph

In Part I, we focus on communication over the AWGN channel. The performance of SPARCs with optimal (least-squares) decoding is analyzed in Chapter 2. Though optimal decoding is infeasible, its performance provides a benchmark for the computationally efficient decoders described in the next chapter. It is shown that SPARCs with optimal encoding achieve the AWGN capacity with an error exponent of the same order as Shannon's random coding ensemble. Similar results are also obtained for SPARCs defined via Bernoulli dictionaries rather than Gaussian ones.

In Chapter 3, we describe three efficient iterative decoders. These decoders generate an updated estimate of the message vector in each iteration based on a test statistic. The first decoder makes hard decisions, decoding a few sections of the message vector β in each iteration. The other two decoders are based on soft-decisions, and generate new estimates of the whole message vector in each iteration. All three efficient decoders are asymptotically capacity-achieving, but the soft-decision decoders have better finite length error performance.

In Chapter 4, we turn our attention to techniques for improving the decoding performance at moderate block lengths. We observe that the power allocation (choice of the non-zero coefficients $\{c_\ell\}$) has a crucial effect on the finite length error performance. We describe an algorithm to determine a good power allocation, provide guidelines on choosing the parameters of the design matrix, and compare the empirical performance with coded modulation using LDPC codes from the WiMAX standard. In Chapter 5, we discuss spatially coupled SPARCs, which consist of several smaller SPARCs chained together in a band-diagonal structure. An attractive feature of spatially coupled SPARCs is that they are asymptotically capacity-achieving and have good finite length performance without requiring a tailored power allocation. Figure 1.2 shows the finite length error rate performance of power allocated SPARCs and spatially coupled SPARCs over an AWGN channel. The figure is discussed in detail in Sec. 5.4.

4

In Part II of the monograph, we use SPARCs for lossy compression with the squared error distortion criterion. In Chapter 6, we analyze compression with optimal (least-squares) encoding, and show that SPARCs attain the optimal rate-distortion function and the optimal excess-distortion exponent for i.i.d. Gaussian sources. We then describe an efficient successive cancellation encoder in Chapter 7, and show that it achieves the optimal Gaussian rate-distortion function, with the probability of excess distortion decaying exponentially in the block length.

In Part III, we design rate optimal coding schemes using SPARCs for a few canonical models in multiuser information theory. In Chapter 8, we show how SPARCs designed for point-to-point AWGN channels can be combined to construct rate-optimal superposition coding schemes for the AWGN broadcast and multiple-access channels. In Chapter 9, we show how to implement random binning using SPARCs. Using this, we can nest the channel coding and source coding SPARCs constructed in Parts I and II to construct rate-optimal schemes for a variety of problems in multiuser information theory. We conclude in Chapter 10 with a discussion of open problems and directions for future work.

Proofs or proof sketches for the main results in a chapter are given at the end of the chapter. The proofs of some intermediate lemmas are omitted, with pointers to the relevant references. The goal is to describe the key technical ideas in the proofs, while not impeding the flow within the chapter.

Part I

AWGN Channel Coding with SPARCs

Chapter 2

Optimal Decoding

In this chapter, we consider sparse regression codes for the additive white Gaussian noise (AWGN) channel, and analyze the performance under optimal (maximum-likelihood) decoding. Though the optimal decoder has computational complexity that grows exponentially with n, its decoding performance sets a benchmark for the efficient decoders discussed in the next chapter. The results in this chapter show that the SPARC error probability with optimal decoding decays exponentially with n for any rate R less than the AWGN channel capacity. In particular, we will see that the error probability bound for SPARCs has the same form as the bound for a Shannon-style random codebook consisting of independent Gaussian codewords [46, 87], but with a weaker constant. We note that a Shannon-style random codebook is infeasible except for very short code lengths as the complexity of encoding and decoding grow exponentially with n.

2.1 Problem set-up

Channel Model   The discrete-time AWGN channel is described by the model

$$y_i = x_i + w_i, \quad i = 1, \ldots, n. \tag{2.1}$$

That is, the channel output y_i at time instant i is the sum of the channel input x_i and the Gaussian noise variable w_i. The random variables w_i, $1 \leq i \leq n$, are i.i.d. $\sim \mathcal{N}(0, \sigma^2)$. There is an average power constraint P on the input: the codeword $x = (x_1, \ldots, x_n)$ should satisfy $\frac{1}{n} \sum_{i=1}^{n} x_i^2 \leq P$. The signal-to-noise ratio $\frac{P}{\sigma^2}$ is denoted by snr.

We wish to use the sparse regression codebook described in Section 1.1 to communicate reliably at any rate $R < \mathcal{C}$, where the channel capacity $\mathcal{C} = \frac{1}{2} \log(1 + \mathsf{snr})$.

Power Allocation   We need to specify the non-zero coefficients $c_1, \ldots, c_L$ in the message vector so as to satisfy the power constraint. Recall that the entries of each codeword $A\beta$ are i.i.d.

9

$\mathcal{N}(0, \frac{1}{n} \sum_{\ell=1}^{L} c_\ell^2)$. In this chapter, we consider the flat power allocation with

$$c_1 = c_2 \ldots = c_L = \sqrt{\frac{nP}{L}}.$$

This choice ensures that for a message $\beta \in \mathcal{B}_{M,L}$, the expected codeword power, given by $\mathbb{E}\|A\beta\|^2/n$, equals P. Using standard large deviations techniques, it can be shown that the distribution of the average codeword power $\|A\beta\|^2/n$ is tightly concentrated around P [15, Appendix B].

In the next two chapters, we will consider different power allocations where the coefficients $c_1, \ldots, c_L$ are not equal to one another. One example is the exponentially decaying allocation, where $c_\ell \propto e^{-\mathcal{C}\ell/L}$, for $\ell \in [L]$. As we will see, such power allocations facilitate computationally feasible decoders that are reliable at rates close to capacity.

Encoding   The encoder splits its stream of input bits into segments of $\log M$ bits each. A length ML message vector β is indexed by L such segments — the decimal equivalent of segment ℓ determines the position of the non-zero coefficient in section ℓ of β. The input codeword is then computed as $x = A\beta$. Note that computing x simply involves adding L columns of A, weighted by the appropriate coefficients.

Maximum Likelihood Decoding   Assuming that the messages are equally likely is equivalent to assuming a uniform prior over $\mathcal{B}_{M,L}$ for the message vector β. Then the decoder that minimizes the probability of message decoding error is the maximum likelihood decoder. We will refer to the maximum likelihood decoder as the optimal decoder. Given the channel output sequence $y = (y_1, \ldots, y_n)$, the optimal decoder produces

$$\hat{\beta}_{\mathsf{opt}} = \underset{\hat{\beta} \in \mathcal{B}_{M,L}}{\arg\min} \|y - A\hat{\beta}\|^2. \tag{2.2}$$

Probability of Decoding Error   A natural performance metric for a SPARC decoder is the *section error rate*, which is the fraction of sections decoded incorrectly. If the true message vector is β and the decoded message vector is $\hat{\beta}$, the section error rate is defined as

$$\mathcal{E}_{\mathsf{sec}} = \frac{1}{L} \sum_{\ell=1}^{L} \mathbf{1}\{\hat{\beta}_\ell \neq \beta_\ell\}, \tag{2.3}$$

where $\beta_\ell, \hat{\beta}_\ell \in \mathbb{R}^M$ denote the ℓth section of $\beta, \hat{\beta}$, respectively. We will first aim to bound the probability of excess section error rate, i.e., the probability of the event $\{\mathcal{E}_{\mathsf{sec}} \geq \epsilon\}$, for $\epsilon > 0$.

Assuming that the mapping that determines the non-zero location within a section for each segment of $\log M$ input bits is generated uniformly at random, a section error will, on average, lead to half the bits corresponding to the section being decoded wrongly. Therefore, when a large number of segments are transmitted, the *bit error rate* of a SPARC decoder will be close to half its section error rate.

Finally, one may also wish to minimize the probability of codeword (or message) error, i.e., $\mathbb{P}(\hat{\beta} \neq \beta)$. For this, one can use a concatenated code with the SPARC as the inner code and an outer Reed-Solomon (RS) code. Later in this chapter (see p. 13), we describe how an RS code of rate $(1 - 2\epsilon)$ can be used to ensure that $\hat{\beta} = \beta$ whenever the section error rate $\mathcal{E}_{\sf sec} < \epsilon$, for any $\epsilon > 0$. With a SPARC of rate R, such a concatenated code has rate $R(1 - 2\epsilon)$ and its probability of codeword error is bounded by $\mathbb{P}(\mathcal{E}_{\sf sec} \geq \epsilon)$. The main result of this chapter, Theorem 2.1, shows that $\mathbb{P}(\mathcal{E}_{\sf sec} \geq \epsilon)$ decays exponentially in n for any $R < \mathcal{C}$.

2.2 Performance of the optimal decoder

The goal is to obtain bounds on the probability of excess section error rate, averaged over all messages and over the space of design matrices. More precisely, for any $\epsilon > 0$, we wish to bound

$$\mathbb{P}(\mathcal{E}_{\sf sec} \geq \epsilon) = \mathbb{E}_{A,\beta} \left[\mathbb{P}(\mathcal{E}_{\sf sec} \geq \epsilon \mid A, \beta) \right]$$
$$= \frac{1}{M^L} \sum_{\beta \in \mathcal{B}_{M,L}} \mathbb{E}_A \left[\mathbb{P}(\mathcal{E}_{\sf sec} \geq \epsilon \mid A, \beta) \right], \tag{2.4}$$

where the subscripts indicate the random variable(s) the expectation is computed over. In (2.4), we note that the probability measure on A is that induced by its i.i.d. $\mathcal{N}(0,1)$ entries. By symmetry, $\mathbb{E}_A \left[\mathbb{P}(\mathcal{E}_{\sf sec} \geq \epsilon \mid A, \beta) \right]$ is the same for all $\beta \in \mathcal{B}_{M,L}$. Therefore we shall obtain bounds for

$$\mathbb{P}_{\beta_0}(\mathcal{E}_{\sf sec} \geq \epsilon) = \mathbb{E}_A \left[\mathbb{P}(\mathcal{E}_{\sf sec} \geq \epsilon \mid A, \beta_0) \right], \tag{2.5}$$

for a fixed message vector $\beta_0 \in \mathcal{B}_{M,L}$.

Preliminaries   We list some facts and definitions that will be used in the bounds.

If $Z, \tilde{Z}$ are jointly Gaussian random variables with means equal to 0, variances equal to 1, and correlation coefficient ρ, then we have the following Chernoff bound for the difference of their squares. For any $\Delta > 0$,

$$\mathbb{P}\left(\frac{1}{2}(Z^2 - \tilde{Z}^2) > \Delta \right) \leq \exp\left(-D(\Delta, 1 - \rho^2) \right), \tag{2.6}$$

where the Cramér-Chernoff large deviation exponent is

$$D(\Delta, 1 - \rho^2) = \max_{\lambda \geq 0} \left\{ \lambda \Delta + \frac{1}{2} \log \left(1 - \lambda^2(1 - \rho^2) \right) \right\}. \tag{2.7}$$

We also define

$$D_1(\Delta, 1 - \rho^2) = \max_{0 \leq \lambda \leq 1} \left\{ \lambda \Delta + \frac{1}{2} \log \left(1 - \lambda^2(1 - \rho^2) \right) \right\}. \tag{2.8}$$

Finally, for $0 \leq \alpha \leq 1$, let

$$\mathcal{C}_\alpha = \frac{1}{2} \log(1 + \alpha\, {\sf snr}). \tag{2.9}$$

Recalling that the capacity $\mathcal{C} = \mathcal{C}_1$, we note that $\mathcal{C}_\alpha - \alpha\mathcal{C}$ is a concave function equal to 0 when α is 0 or 1, and strictly positive in between.

Error probability bounds The first result is a non-asymptotic bound on the probability of excess section error rate defined in (2.5).

Proposition 2.1. *[16, Eq. (24)] For any $\beta_0 \in \mathcal{B}_{M,L}$ and $\epsilon > 0$,*

$$\mathbb{P}_{\beta_0}(\mathcal{E}_{sec} \geq \epsilon) \leq \sum_{\ell = \epsilon L}^{L} \min\left\{err_1(\ell/L),\ err_2(\ell/L)\right\}, \tag{2.10}$$

where for $0 < \alpha \leq 1$, the functions $err_1(\alpha)$ and $err_2(\alpha)$ are defined as follows.

$$err_1(\alpha) = \binom{L}{\alpha L} \exp\left(-nD_1\left(\mathcal{C}_\alpha - \alpha R, \frac{\alpha\, snr}{1 + \alpha\, snr}\right)\right), \tag{2.11}$$

$$err_2(\alpha) = \min_{t_\alpha \in [0, \mathcal{C}_\alpha - \alpha R]} err_2(\alpha, t_\alpha), \tag{2.12}$$

where

$$err_2(\alpha, t_\alpha) = \binom{L}{\alpha L} \exp\left(-nD_1\left(\mathcal{C}_\alpha - \alpha R - t_\alpha, \frac{\alpha(1-\alpha)\, snr}{1 + \alpha\, snr}\right)\right)$$
$$+ \exp\left(-nD\left(t_\alpha, \frac{\alpha^2\, snr}{1 + \alpha^2\, snr}\right)\right). \tag{2.13}$$

The proof of the proposition is given in Section 2.4.1.

The bound in (2.10) can be computed numerically given the rate R and the SPARC parameters (M, L). The function $err_1(\alpha)$ gives the better bound for α close to 0, while $err_2(\alpha)$ is better for α close to 1.

The next result simplifies the non-asymptotic bound and shows that the probability of excess section error rate decays exponentially in n, with the exponent depending on the gap from capacity $\Delta = \mathcal{C} - R$. First, a few definitions that are needed to state the result.

For $x > 0$, let

$$g(x) = \sqrt{1 + 4x^2} - 1. \tag{2.14}$$

It follows that

$$g(x) \geq \min\{\sqrt{2}x, x^2\} \quad \text{for all} \quad x \geq 0. \tag{2.15}$$

Next, let

$$w(snr) = \frac{snr}{2(1 + snr)^2\sqrt{4 + snr^3/(1 + snr)}}. \tag{2.16}$$

Finally, let $a_L^*(snr)$ be defined as

$$a_L^*(snr) = \max_{\alpha \in \{\frac{1}{L}, \ldots, 1 - \frac{1}{L}\}} \frac{R \log\left(\frac{L}{L\alpha}\right)}{D_1\left(\mathcal{C}_\alpha - \alpha\mathcal{C}, \frac{\alpha(1-\alpha)snr}{1 + \alpha snr}\right) L \log L}, \tag{2.17}$$

where the D_1 is defined in (2.8). The behavior of $a_L^*(snr)$ as $L \to \infty$ is described in Remark 2.2.

12

Theorem 2.1. *[16] Assume that $M = L^{\mathsf{a}}$, where $\mathsf{a} \geq \mathsf{a}_L^*(snr)$, and that the gap from capacity $\Delta = (\mathcal{C} - R)$ is strictly positive. Then, for any $\epsilon > 0$, the section error rate $\mathcal{E}_{sec}$ of the optimal decoder satisfies*

$$\mathbb{P}\left(\mathcal{E}_{sec} \geq \epsilon\right) = e^{-nE(\epsilon, R)}, \tag{2.18}$$

with

$$E(\epsilon, R) \geq h(\epsilon, \Delta) - \frac{\log 2L}{n}, \tag{2.19}$$

where

$$h(\epsilon, \Delta) = \min\left\{\epsilon \Delta w(snr), \frac{1}{4}g\left(\frac{\Delta}{2\sqrt{snr}}\right)\right\}. \tag{2.20}$$

The proofs of the theorem is based on Proposition 2.1, and is given in Section 2.4.2.

Remark 2.1. *The lower bound on $g(x)$ in (2.15) implies that the function $h(\epsilon, \Delta)$ can be bounded from below as*

$$h(\epsilon, \Delta) \geq \min\left\{\epsilon \Delta w(snr), \frac{\Delta}{\sqrt{32snr}}, \frac{\Delta^2}{16snr}\right\},$$

revealing that the exponent is, up to a constant, of the form $\min\{\epsilon \Delta, \Delta^2\}$.

An improved lower bound on the exponent $E(\epsilon, R)$ is obtained in [16, Appendix C]. This lower bound replaces the function $h(\epsilon, \Delta)$ with a larger function $\tilde{h}(\epsilon, \Delta)$, and shows that the exponent is of the form $\min\{\epsilon, \Delta^2\}$.

Remark 2.2. *The parameter $\mathsf{a}_L^*(snr)$ approaches the following limiting value as $L \to \infty$. Let v^* near 15.8 be the solution to the equation $(1 + v^*)\log(1 + v^*) = 3v^*$. Then [16, Lemma 5] shows that*

$$\lim_{L \to \infty} \mathsf{a}_L^*(snr) = \begin{cases} \frac{8R\,snr\,(1+snr)}{[(1+snr)\log(1+snr) - snr]^2} & \text{for } snr < v^*, \\ \frac{2R(1+snr)}{[(1+snr)\log(1+snr) - 2\,snr]} & \text{for } snr \geq v^*. \end{cases} \tag{2.21}$$

Taking the upper bound of $\mathcal{C}$ for R, it can be shown that the above limit is approximately $16/snr^2$ for small values of snr, and 1 for large snr.

Probability of message error Using a suitable outer code, the bound on the probability of excess section error in Theorem 2.1 can be translated into a bound on the probability of message error, i.e., $\mathbb{P}(\hat{\beta} \neq \beta)$.

Consider a concatenated code with a SPARC of rate R as the inner code, and a Reed-Solomon (RS) outer code, chosen as follows. For simplicity, assume that $M = 2^m$. We consider a systematic (n_{out}, k_{out}) RS code with symbols in $GF(2^m)$. From the theory of RS codes [22, 80], we can take

$$n_{out} = M, \quad k_{out} = \lceil (1 - \epsilon)M \rceil. \tag{2.22}$$

to obtain an RS code with minimum distance

$$d_{\mathsf{RS}} = M - \lceil (1 - \epsilon)M \rceil + 1 \quad \text{symbols.} \tag{2.23}$$

The information bits are encoded into the SPARC codeword as follows. First consider the case where $L = M$. Here, the RS encoder maps $k_{out}m$ information bits (k_{out} symbols) into a length L RS codeword. Since each SPARC section has M columns, each symbol of the RS encoder represents the index for one section of the SPARC. For the case where $L < M$, we can use the same procedure by setting the first $(M - L)$ symbols of the systematic RS codeword to 0.

From (2.23), the number of symbol errors that this code is guaranteed to correct in a length n_{out} codeword is

$$\left\lfloor \frac{d_{\mathsf{RS}}}{2} \right\rfloor \geq \lfloor \epsilon M \rfloor.$$

Therefore, the decoded message $\hat{\beta}$ equals the transmitted one β whenever the optimal SPARC decoder makes no more than $\lfloor \epsilon M \rfloor$ section errors. Therefore, the probability of message error for the concatenated code is bounded by the RHS of (2.18). From (2.22), the rate of the concatenated code is at least $R(1 - 2\epsilon)$, where R is the rate of the SPARC.

We therefore have the following result.

Proposition 2.2. *[16] Consider a SPARC with rate $R < \mathcal{C}$, with parameters (M, L) satisfying the assumptions of Theorem 2.1. Then for any $\epsilon > 0$, through concatenation with an outer RS code, one obtains a code of rate $R(1 - 2\epsilon)$ with message error probability bounded by $e^{-nE(\epsilon, R)}$. Here $E(\epsilon, R)$ is the exponent from Theorem 2.1, which can be bounded from below as in (2.19).*

Remark 2.3. *Consider the regime where the SPARC rate R is made to approach $\mathcal{C}$ as $R = C - \Delta_n$. Let Δ_n tend to zero at a rate slower than $1/\sqrt{n}$, e.g., $\frac{1}{n^{1/4}}$ or $\frac{1}{\log n}$). Then choosing $\epsilon = \Delta_n$, Proposition 2.2 shows that we have a code whose overall rate is $(\mathcal{C} - \Delta_n)(1 - \Delta_n)$ and whose probability of message error decays as $\exp(-\kappa n \Delta_n^2)$, where κ is a universal positive constant.*

2.3 Performance with i.i.d. Bernoulli dictionaries

SPARCs defined via an i.i.d. Gaussian design matrix are not suitable for practical implementation, especially for large code lengths. Large Gaussian design matrix have prohibitive storage complexity as the entries will span a large range of real numbers which need to be stored with high precision. To reduce the storage requirement, one could define the SPARC via a Bernoulli design matrix entries are chosen uniformly at random from the set $\{1, -1\}$. As before, the set of valid message vectors is $\mathcal{B}_{M,L}$, i.e., β which have one non-zero in each of the L sections. In this section, we consider Bernoulli-defined SPARCs with equal power allocation, i.e., each non-zero entry of β equals $\sqrt{P/L}$. Each entry of the codeword is therefore a sum of L i.i.d. random variables, each drawn uniformly from $\left\{ \sqrt{P/L}, -\sqrt{P/L} \right\}$. Therefore, by the central limit theorem, each codeword entry converges in distribution to an i.i.d. $\mathcal{N}(0, P)$ random variable.

The performance of Bernoulli dictionaries with optimal decoding was analyzed by Takeishi et al. [105, 106]. The main result, stated below, gives an error probability bound that is almost identical to the one in Theorem 2.1 for the Gaussian case, except for a slightly weaker exponent.

Theorem 2.2. *[106] With the same assumptions and notation as in Theorem 2.1, the section error*

rate of a SPARC defined with a Bernoulli dictionary satisfies

$$\mathbb{P}\left(\mathcal{E}_{sec} \geq \epsilon\right) = e^{-nE(\epsilon, R)}, \tag{2.24}$$

with

$$E(\epsilon, R) \geq h(\epsilon, \Delta) - \iota(L), \tag{2.25}$$

where $\iota(L) = O(1/\sqrt{L})$.

We note that the only difference from result for the Gaussian case is that in the lower bound for $E(\epsilon, R)$, the $\log(2L)/n$ term is now replaced with $\iota(L) = O(1/\sqrt{L})$. A proof sketch of the proposition is given in Section 2.4.3.

2.4 Proofs

2.4.1 Proof of Proposition 2.1

We obtain (2.10) by proving that for $\ell \in \{1, \ldots, L\}$,

$$\mathbb{P}_{\beta_0}(\mathcal{E}_{sec} = \ell/L) \leq \mathsf{err}_1(\ell/L), \tag{2.26}$$

$$\mathbb{P}_{\beta_0}(\mathcal{E}_{sec} = \ell/L) \leq \mathsf{err}_2(\ell/L), \tag{2.27}$$

where $\mathsf{err}_1(\cdot)$ and $\mathsf{err}_2(\cdot)$ are defined in the statement of the proposition.

For any $\beta \in \mathcal{B}_{M,L}$, let $\mathcal{S}(\beta) = \{j : \beta_j = 1\}$ denote the set of non-zero indices. Let $\mathcal{S}^* = \mathcal{S}(\beta_0)$ denote the set of non-zero indices for the true message vector β_0, and let $\mathcal{S}$ denote the set of non-zero indices in the decoded message vector When there are ℓ section errors, the set $\mathcal{S}$ differs from $\mathcal{S}^*$ in exactly ℓ elements. Letting $X_{\mathcal{S}} = A\beta_{\mathcal{S}}$ and $X_{\mathcal{S}^*} = A\beta_{\mathcal{S}^*} = A\beta_0$, the ML decoder decodes $\mathcal{S}$ only when the received vector Y satisfies $\|Y - X_{\mathcal{S}}\|^2 \leq \|Y - X_{\mathcal{S}^*}\|^2$, or equivalently, when $T(\mathcal{S}) \leq 0$, where

$$T(\mathcal{S}) = \frac{1}{2n}\left[\frac{\|Y - X_{\mathcal{S}}\|^2}{\sigma^2} - \frac{\|Y - X_{\mathcal{S}^*}\|^2}{\sigma^2}\right]. \tag{2.28}$$

The analysis proceeds by obtaining a bound for $\mathbb{P}_{\beta_0}(T(\mathcal{S}) \leq 0)$ that holds for each choice of $\mathcal{S}$. Noting that there are $\binom{L}{\ell}M^\ell$ choices for $\mathcal{S}$, the natural way to combine these is via a union bound:

$$\mathbb{P}_{\beta_0}(\mathcal{E}_{sec} = \ell/L) = \binom{L}{\ell}M^\ell \mathbb{P}_{\beta_0}(T(\mathcal{S}) \leq 0).$$

However, such a union bound gives a result weaker than that of Proposition 2.1. Therefore, we obtain (2.26) and (2.27) by decomposing $T(\mathcal{S})$ in two different ways and using a modified union bound.

Proof of (2.26) [16, Lemma 3]: Let $\mathcal{S}_1 = \mathcal{S} \cap \mathcal{S}^*$ and $\mathcal{S}_2 = \mathcal{S} - \mathcal{S}_1$ denote, respectively, the intersection and difference between sets $\mathcal{S}$ and $\mathcal{S}^*$. With $\alpha = \ell/L$, the sizes of $\mathcal{S}_2$ and $\mathcal{S}_1$ are αL and $L - \alpha L$, respectively. With this notation, the probability of the event $\{\mathcal{E}_{\mathsf{sec}} = \ell/L\}$ can be bounded as follows. For any $\lambda > 0$, the indicator of the event satisfies

$$\mathbf{1}\{\mathcal{E}_{\mathsf{sec}} = \ell/L\} \leq \sum_{\mathcal{S}_1} \left(\sum_{\mathcal{S}_2} e^{-nT(\mathcal{S})} \right)^{\lambda}. \tag{2.29}$$

We decompose the test statistic $T(\mathcal{S})$ as $T_1 + T_2$, where

$$T_1 = \frac{1}{2n} \left[\frac{\|Y - X_{\mathcal{S}_1}\|^2}{\sigma^2 + \alpha P} - \frac{\|Y - X_{\mathcal{S}^*}\|^2}{\sigma^2} \right], \tag{2.30}$$

and

$$T_2 = \frac{1}{2n} \left[\frac{\|Y - X_{\mathcal{S}}\|^2}{\sigma^2} - \frac{\|Y - X_{\mathcal{S}_1}\|^2}{\sigma^2 + \alpha P} \right]. \tag{2.31}$$

Observing that T_1 depends only on the indices in $\mathcal{S}^*$ (and not on those in $\mathcal{S}_2$), we take expectations on both sides of (2.29) to write

$$\begin{aligned}
\mathbb{P}_{\beta_0}(\mathcal{E}_{\mathsf{sec}} = \ell/L) &\leq \sum_{\mathcal{S}_1} \mathbb{E} e^{-n\lambda T_1(\mathcal{S}_1)} \, \mathbb{E}_{X_{\mathcal{S}_2}|Y,X_{\mathcal{S}_1},X_{\mathcal{S}^*}} \left(\sum_{\mathcal{S}_2} e^{-nT_2(\mathcal{S})} \right)^{\lambda} \\
&\overset{(a)}{\leq} \sum_{\mathcal{S}_1} \mathbb{E} e^{-n\lambda T_1(\mathcal{S}_1)} \left[\sum_{\mathcal{S}_2} \mathbb{E}_{X_{\mathcal{S}_2}} e^{-nT_2(\mathcal{S})} \right]^{\lambda} \\
&\overset{(b)}{=} \sum_{\mathcal{S}_1} \mathbb{E} e^{-n\lambda T_1(\mathcal{S}_1)} \left[\sum_{\mathcal{S}_2} \left(1 + \frac{\alpha P}{\sigma^2} \right)^{-\frac{n}{2}} \right]^{\lambda}, \\
&\overset{(c)}{\leq} \sum_{\mathcal{S}_1} \mathbb{E} e^{-n\lambda T_1(\mathcal{S}_1)} \, e^{-n\lambda(\mathcal{C}_\alpha - \alpha R)}. \tag{2.32}
\end{aligned}$$

where (a) is obtained using Jensen's inequality (arranging for λ to be not more than 1), and noting that $X_{\mathcal{S}_2}$ is independent of $(Y, X_{\mathcal{S}_1}, X_{\mathcal{S}^*})$. Step (b) is obtained by writing

$$e^{-nT_2(\mathcal{S})} = \exp\left(-\frac{1}{2} \left[\frac{\|Y - X_{\mathcal{S}_1} - X_{\mathcal{S}_2}\|^2}{\sigma^2} - \frac{\|Y - X_{\mathcal{S}_1}\|^2}{\sigma^2 + \alpha P} \right] \right),$$

and evaluating the expectation with respect to $X_{\mathcal{S}_2}$, which is i.i.d. $\sim \mathcal{N}(0, \alpha P)$. For step (c), we observe that the sum over $\mathcal{S}_2$ involves at most $M^\ell = e^{nR\alpha}$ terms, and use the definition of $\mathcal{C}_\alpha$ from (2.9).

We note from (2.30) that $T_1(\mathcal{S}_1)$ is distributed as

$$T_1(\mathcal{S}_1) \overset{d}{=} \frac{1}{2n} \sum_{i=1}^{n} \left(Z_i^2 - \tilde{Z}_i^2 \right),$$

where each pair $(Z_i, \tilde{Z}_i)$ is bivariate Gaussian with squared correlation equal to $1/(1 + \alpha\mathsf{snr})$. The pairs are i.i.d. for $1 \leq i \leq n$. Using this, the expectation in (2.32) is found to be

$$\mathbb{E} e^{-n\lambda T_1(\mathcal{S}_1)} = (1 - \lambda^2 \alpha\mathsf{snr}/(1 + \alpha\mathsf{snr}))^{-n/2}.$$

The proof is completed by using this in (2.32), noting that the sum over $\mathcal{S}_1$ has $\binom{L}{L\alpha}$ terms, and optimizing the bound over $\lambda \in [0, 1]$.

16

Proof of (2.27) [16, Lemma 4]: For any $\mathcal{S}$ which differs from $\mathcal{S}^*$ in ℓ sections, we decompose the test statistic in (2.28) as $T(\mathcal{S}) = \tilde{T}(\mathcal{S}) + T^*$, where

$$\tilde{T}(\mathcal{S}) = \frac{1}{2n} \left[\frac{\|Y - X_{\mathcal{S}}\|^2}{\sigma^2} - \frac{\|Y - (1-\alpha)X_{\mathcal{S}^*}\|^2}{\sigma^2 + \alpha^2 P} \right], \qquad (2.33)$$

$$T^* = \frac{1}{2n} \left[\frac{\|Y - (1-\alpha)X_{\mathcal{S}^*}\|^2}{\sigma^2 + \alpha^2 P} - \frac{\|Y - X_{\mathcal{S}^*}\|^2}{\sigma^2} \right]. \qquad (2.34)$$

Let $t_\alpha \in [0, \mathcal{C}_\alpha - \alpha R]$. Then,

$$\mathbb{P}_{\beta_0}(\mathcal{E}_{\text{sec}} = \ell/L) \le \mathbb{P}_{\beta_0}(\exists \mathcal{S} : \tilde{T}(\mathcal{S}) \le t_\alpha) + \mathbb{P}_{\beta_0}(T^* \le -t_\alpha). \qquad (2.35)$$

We note that T^* does not depend on $\mathcal{S}$: it is a mean zero average of the difference of squared Gaussian random variables, with squared correlation $1/(1 + \alpha^2\text{snr})$. The second term on the RHS of (2.35) can therefore be bounded via a Chernoff bound (as in (2.6)) to obtain the second term in (2.13).

The analysis of the first term in (2.35) is very similar to the proof of (2.26) above. We write $\tilde{T}(\mathcal{S}) = \tilde{T}_1 + \tilde{T}_2$, where

$$\tilde{T}_1 = \frac{1}{2n} \left[\frac{\|Y - X_{\mathcal{S}_1}\|^2}{\sigma^2 + \alpha P} - \frac{\|Y - (1-\alpha)X_{\mathcal{S}^*}\|^2}{\sigma^2 + \alpha P} \right],$$

$$\tilde{T}_2 = \frac{1}{2n} \left[\frac{\|Y - X_{\mathcal{S}}\|^2}{\sigma^2} - \frac{\|Y - X_{\mathcal{S}_1}\|^2}{\sigma^2 + \alpha P} \right]. \qquad (2.36)$$

The key difference between $\tilde{T}_1$ and T_1 (defined in (2.30)) is that the two standard normals have a higher correlation coefficient in $\tilde{T}_1$. Indeed, the squared correlation coefficient between the two standard normals in (2.36) is $\rho_\alpha^2 = (1 + \alpha^2\text{snr})/(1 + \alpha\text{snr})$, and the moment generating function is found to be

$$\mathbb{E}e^{-n\lambda\tilde{T}_1(\mathcal{S}_1)} = (1 - \lambda^2\alpha(1-\alpha)\text{snr}/(1 + \alpha\text{snr}))^{-n/2}.$$

Following steps similar to (2.32) yields the second term in (2.13).

This proves (2.27), and therefore Proposition 2.1. □

2.4.2 Proof of Theorem 2.1

To prove Theorem 2.1, we use the following weaker bound implied by Proposition 2.1: $\mathbb{P}_{\beta_0}(\mathcal{E}_{\text{sec}} \ge \epsilon) \le \sum_{\ell=\epsilon L}^{L} \text{err}_2(\ell/L)$, where $\text{err}_2(\cdot)$ is defined in (2.13). Let

$$\Delta_\alpha = \mathcal{C}_\alpha - \alpha R - t_\alpha, \quad \tilde{\Delta}_\alpha = \mathcal{C}_\alpha - \alpha\mathcal{C}, \quad 1 - \rho_\alpha^2 = \frac{\alpha(1-\alpha)\text{snr}}{1 + \alpha\text{snr}}. \qquad (2.37)$$

Noting that $\Delta_\alpha = \tilde{\Delta}_\alpha + \alpha(\mathcal{C} - R) - t_\alpha$, the idea is to cancel the combinatorial coefficient $\binom{L}{L\alpha}$ using $\exp(-nD_1(\tilde{\Delta}_\alpha, 1-\rho_\alpha^2))$, and produce an exponentially small error probability using $\exp(-n[D_1(\Delta_\alpha, 1-\rho_\alpha^2) - D_1(\tilde{\Delta}_\alpha, 1-\rho_\alpha^2)])$.

17

The derivative of $D_1(\Delta, 1 - \rho_\alpha^2)$ with respect to Δ, denoted by $D_1'(\Delta)$, is equal to

$$D_1'(\Delta) = \begin{cases} \frac{2\Delta}{(1-\rho_\alpha^2)(1+\sqrt{1+4\Delta^2/(1-\rho_\alpha^2)})} & \text{if } \Delta < \frac{1-\rho_\alpha^2}{\rho_\alpha^2}, \\ 1 & \text{otherwise.} \end{cases} \tag{2.38}$$

Since the derivative is non-decreasing in Δ, using a first-order Taylor expansion we deduce

$$D_1(\Delta_\alpha, 1 - \rho_\alpha^2) \geq D_1(\tilde{\Delta}_\alpha, 1 - \rho_\alpha^2) + (\Delta_\alpha - \tilde{\Delta}_\alpha)D_1'(\tilde{\Delta}_\alpha). \tag{2.39}$$

Then using the definition of err_2 in (2.13), we have for $\frac{1}{L} \leq \alpha \leq 1 - \frac{1}{L}$,

$$\begin{aligned}
\mathsf{err}_2(\alpha) &\leq \exp\left(-nD\left(t_\alpha, \frac{\alpha^2\,\mathsf{snr}}{1+\alpha^2\,\mathsf{snr}}\right)\right) \\
&\quad + \binom{L}{\alpha L}\exp(-nD_1(\tilde{\Delta}_\alpha, 1-\rho_\alpha^2))\exp(-n(\alpha(\mathcal{C}-R)-t_\alpha)D_1'(\tilde{\Delta}_\alpha)) \\
&\leq \exp\left(-nD\left(t_\alpha, \frac{\alpha^2\,\mathsf{snr}}{1+\alpha^2\,\mathsf{snr}}\right)\right) + \exp(-n(\alpha(\mathcal{C}-R)-t_\alpha)D_1'(\tilde{\Delta}_\alpha)) \tag{2.40}
\end{aligned}$$

where the last inequality is obtained by using the relation $nR = L\log M = \mathsf{a}L\log L$, and the fact that $\mathsf{a} \geq \mathsf{a}_L^*(\mathsf{snr})$ (see (2.17)). Choosing $t_\alpha = \alpha(\mathcal{C} - R)/2$, we obtain

$$\begin{aligned}
\mathsf{err}_2(\alpha) &\leq \exp\left(-nD\left(\frac{\alpha(\mathcal{C}-R)}{2}, \frac{\alpha^2\,\mathsf{snr}}{1+\alpha^2\,\mathsf{snr}}\right)\right) + \exp\left(\frac{-n\alpha(\mathcal{C}-R)D_1'(\tilde{\Delta}_\alpha)}{2}\right) \\
&\stackrel{(a)}{\leq} \exp\left(-nD\left(\frac{\alpha(\mathcal{C}-R)}{2}, \frac{\alpha^2\,\mathsf{snr}}{1+\alpha^2\,\mathsf{snr}}\right)\right) + \exp\left(-n\alpha(\mathcal{C}-R)w(\mathsf{snr})\right) \\
&\stackrel{(b)}{\leq} \exp\left(\frac{-n}{4}g\left(\frac{(\mathcal{C}-R)\sqrt{1+\alpha^2\mathsf{snr}}}{2\sqrt{\mathsf{snr}}}\right)\right) + \exp\left(-n\alpha(\mathcal{C}-R)w(\mathsf{snr})\right) \tag{2.41}
\end{aligned}$$

where the function g is defined in (2.14). In the above, inequality (a) is obtained using the lower bound $D_1'(\tilde{\Delta}_\alpha) \geq 2w(\mathsf{snr})$, with $w(\mathsf{snr})$ being defined in (2.16). This lower bound is obtained by using the definition of $1 - \rho_\alpha^2$ from (2.37) and the lower bound $\tilde{\Delta}_\alpha \geq \frac{\mathsf{snr}}{4(1+\mathsf{snr})^2}\alpha(1-\alpha)$ in the expression for $D_1'(\tilde{\Delta}_\alpha)$ in (2.38). Inequality (b) is obtained using the following lower bound [16, Lemma 6]:

$$D(x, 1 - \rho^2) \geq \frac{1}{4}g\left(\sqrt{1 + \frac{4x^2}{1 - \rho^2}} - 1\right).$$

Finally, using (2.41) in (2.10) and noting that there are at most L terms in the sum, we deduce

$$\begin{aligned}
\mathbb{P}_{\beta_0}(\mathcal{E}_{\mathsf{sec}} \geq \epsilon) &\leq 2L\exp\left(-n\min\left\{\epsilon\Delta w(\mathsf{snr}), \frac{1}{4}g\left(\frac{\Delta}{2\sqrt{\mathsf{snr}}}\right)\right\}\right) \\
&= \exp\left(-h(\epsilon, \Delta) - \frac{\log 2L}{n}\right),
\end{aligned}$$

where $\Delta = (\mathcal{C} - R)$, and $h(\epsilon, \Delta)$ is defined in (2.20). This completes the proof of Theorem 2.1.

2.4.3 Proof sketch of Theorem 2.2

We will prove the theorem via the following bound similar to Proposition 2.1:

$$\mathbb{P}_{\beta_0}(\mathcal{E}_{\mathsf{sec}} \geq \epsilon) \leq \sum_{\ell=\epsilon L}^{L} \mathsf{err}_2'(\ell/L) \tag{2.42}$$

where $\mathsf{err}_2'(\cdot)$ is defined as follows. For $0 < \alpha \leq 1$,

$$\mathsf{err}_2'(\alpha) = \min_{t_\alpha \in [0, C_\alpha - \alpha R]} \mathsf{err}_2'(\alpha, t_\alpha), \tag{2.43}$$

where

$$\mathsf{err}_2'(\alpha, t_\alpha) = \binom{L}{\alpha L} \exp\left(-n D_1\left(C_\alpha - \alpha R - t_\alpha, \frac{\alpha(1-\alpha)\,\mathsf{snr}}{1 + \alpha\,\mathsf{snr}}\right) - \iota_1\right)$$
$$+ \exp\left(-n D\left(t_\alpha, \frac{\alpha^2\,\mathsf{snr}}{1 + \alpha^2\,\mathsf{snr}} - \iota_2\right)\right). \tag{2.44}$$

Here $\iota_1 = O(1/\sqrt{L})$ and $\iota_2 = O(1/L)$.

The only difference between $\mathsf{err}_2'(\cdot)$ and $\mathsf{err}_2(\cdot)$ defined in (2.13) is the presence of ι_1 and ι_2 in the former. Using the bound in (2.42), Theorem 2.2 can be established using steps similar to those used for Theorem 2.1 in Sec. 2.4.2.

We now sketch the proof of (2.42). The proof hinges on two key lemmas. The first uniformly bounds the ratio between a binomial pmf and a Gaussian with the same mean and variance.

Lemma 2.3. [105] Let $\phi(x; \mu, \sigma^2)$ denote the normal density with mean μ and variance σ^2. Then for any $\ell \in \mathbb{N}$,

$$\max_{k \in \{0,1,\ldots,\ell\}} \frac{\binom{\ell}{k} 2^{-\ell}}{\phi(k; \ell/2, \ell/4)} \leq \exp(\varphi(\ell)),$$

where $\varphi(\ell) \leq 5/\ell$ for $\ell \geq 1000$.

The next two lemmas give bounds on the ratio of certain Reimann sums to the corresponding integrals.

Lemma 2.4. [106] For $n \in \mathbb{N}$, let $h = 2/\sqrt{n}$ and $x_k = -\sqrt{n} + \frac{2k}{\sqrt{n}}$ for $k = 0, 1, \ldots, n$. For $\mu \in \mathbb{R}$ and $s > 0$, define

$$I_d = h \sum_{k=0}^{n} \exp\left\{-\frac{s^2}{2}(x_k - \mu)^2\right\},$$
$$I_c = \int_{-\infty}^{\infty} \exp\left\{-\frac{s^2}{2}(x - \mu)^2\right\} dx.$$

Then

$$I_d \leq \left(1 + \frac{\eta s^2}{n}\right) I_c,$$

where $\eta = 3/\sqrt{8\pi e} \leq 0.37$.

Lemma 2.5. *For $n, n' \in \mathbb{N}$, let $h = 2/\sqrt{n}$, $\mathcal{X} = \{-\sqrt{n} + \frac{2k}{\sqrt{n}} \mid k = 0, 1, \ldots, n\}$, and let $h' = 2/\sqrt{n'}$, $\mathcal{X} = \{-\sqrt{n'} + \frac{2k}{\sqrt{n'}} \mid k = 0, 1, \ldots, n\}$.*

(a) For a two-dimensional vector $\mathbf{x} = [x_1, x_2]^T \in \mathbb{R}^2$ and a 2×2 positive definite matrix B, define

$$I_d = \int_{\mathbb{R}} h \sum_{x_1 \in \mathcal{X}} \exp\left\{-\mathbf{x}^T B \mathbf{x}/2\right\} dx_2,$$

$$I_c = \int_{\mathbb{R}^2} \exp\left\{-\mathbf{x}^T B \mathbf{x}/2\right\} d\mathbf{x}.$$

Then $I_d \leq \left(1 + \frac{\eta B_{11}}{n}\right) I_c$, where η is defined in Lemma 2.4, and B_{ij} denotes the (i, j)th element of the matrix B.

(b) For a three-dimensional vector $\mathbf{x} = [x_1, x_2, x_3]^T \in \mathbb{R}^3$ and a 3×3 positive definite matrix B, define

$$I_d = \int_{\mathbb{R}} h\, h' \sum_{x_1 \in \mathcal{X}_1} \sum_{x_1 \in \mathcal{X}_2} \exp\left\{-\mathbf{x}^T B \mathbf{x}/2\right\} dx_3,$$

$$I_c = \int_{\mathbb{R}^3} \exp\left\{-\mathbf{x}^T B \mathbf{x}/2\right\} d\mathbf{x}.$$

Then $I_d \leq \left(1 + \frac{\eta B_{11}}{n}\right)\left(1 + \frac{\eta B_{22}}{n'}\right) I_c$.

The proof is along the lines of that of (2.27) on p.17. We have

$$\mathbb{P}_{\beta_0}(\mathcal{E}_{\mathsf{sec}} = \ell/L) \leq \mathbb{P}_{\beta_0}(\exists \mathcal{S} : \tilde{T}(\mathcal{S}) \leq t_\alpha) + \mathbb{P}_{\beta_0}(T^* \leq -t_\alpha). \tag{2.45}$$

with $\tilde{T}(\mathcal{S})$ and T^* defined as in (2.33)-(2.34). Using a Chernoff bound, the second term can be bounded as

$$\mathbb{P}_{\beta_0}(T^* \leq -t_\alpha) \leq e^{-n\lambda t_\alpha} \mathbb{E}_{Y, X_{\mathcal{S}_*}} e^{-n\lambda T^*}. \tag{2.46}$$

The moment generating function can be written as

$$\mathbb{E}_{Y, X_{\mathcal{S}_*}} e^{-n\lambda T^*} = \left[\mathbb{E}_{Z_1, \tilde{Z}_1} e^{\lambda(Z_1^2 - \tilde{Z}_1^2)/2}\right]^n, \tag{2.47}$$

where

$$Z_1 \sim \mathcal{N}(0, 1), \qquad \tilde{Z}_1 = \frac{(\sigma Z + \alpha\sqrt{P}\, W_1)}{\sqrt{\sigma^2 + \alpha^2 P}}.$$

Here W_1 (independent of Z_1) is the sum of L independent equiprobable ± 1 random variables, normalized to have unit variance. If W_1 was Gaussian, then the moment generating function in (2.47) would be exactly equal to $(1 - \lambda^2\alpha^2\mathsf{snr}/(1 + \alpha^2\mathsf{snr}))^{-n/2}$. For the W_1 arising from a Bernoulli dictionary, using Lemmas 2.3, 2.4 and 2.5, it can be shown that

$$\mathbb{E}_{Y, X_{\mathcal{S}_*}} e^{-n\lambda T^*} \leq \left(\frac{e^{t_2}}{1 - \lambda^2\alpha^2\mathsf{snr}/(1 + \alpha^2\mathsf{snr})}\right)^n.$$

20

Using this in (2.46) yields the second term in (2.44).

For the first term in (2.45), we write $\tilde{T}(\mathcal{S}) = \tilde{T}_1 + \tilde{T}_2$ where $\tilde{T}_1$ and $\tilde{T}_2$ are defined in (2.36). Then, using steps similar to (2.32), we obtain

$$\mathbb{P}_{\beta_0}(\exists\,\mathcal{S}:\tilde{T}(\mathcal{S}) \le t_\alpha) \le e^{nt_\alpha} \sum_{\mathcal{S}_1} \mathbb{E}_{Y,X_{\mathcal{S}^*}} e^{-n\lambda T_1(\mathcal{S}_1)} \left[\sum_{\mathcal{S}_2} \mathbb{E}_{X_{\mathcal{S}_2}} e^{-nT_2(\mathcal{S})}\right]^\lambda. \tag{2.48}$$

Again, using Lemmas 2.3, 2.4 and 2.5, the two moment generating functions in (2.48) can be bounded to yield the first term in (2.44). The details of the computation can be found in [105, Section III.C] and [106].

Chapter 3

Computationally Efficient Decoding

In this chapter, we will discuss computationally efficient decoders for SPARCs over the AWGN channel. The goal is to design and analyze feasible capacity-achieving decoders whose complexity is polynomial in the code length n, in contrast to the infeasible maximum-likelihood decoder.

The channel model and the encoding procedure are as described in Sec. 2.1. The first idea for designing a efficient decoder is to use a decaying power allocation across sections. As shown in Fig. 3.1, the non-zero coefficients in the message vector β are

$$c_1 = \sqrt{nP_1}, \; c_2 = \sqrt{nP_2}, \ldots, c_L = \sqrt{nP_L}.$$

Without loss of generality, we assume that the power allocation is non-increasing across sections, i.e., $P_1 \geq P_2 \ldots \geq P_L$. Denoting the column of A corresponding to the ℓth non-zero entry of β by A_{i_ℓ}, for $\ell \in [L]$, the received sequence $y \in \mathbb{R}^n$ is

$$y = \sqrt{nP_1}A_{i_1} + \sqrt{nP_2}A_{i_2} + \ldots + \sqrt{nP_L}A_{i_L} + w. \tag{3.1}$$

The decoding task is to recover the non-zero locations $i_1, \ldots, i_L$. The idea of power allocation is to facilitate an iterative decoder that first decodes (either exactly or approximately) the sections
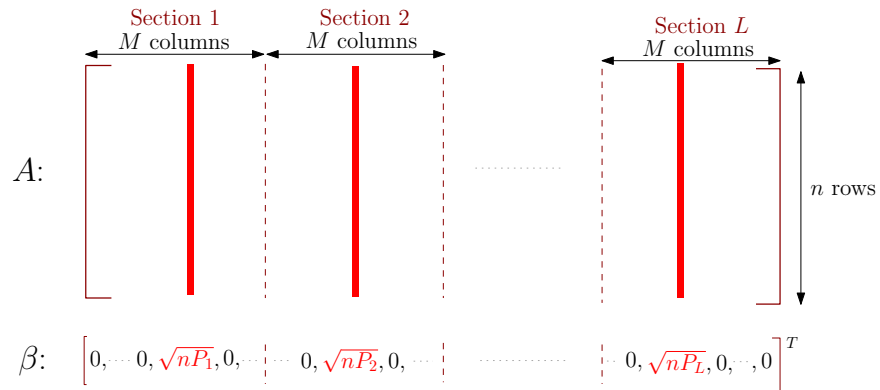

Figure 3.1: A Gaussian sparse regression codebook with power allocation. The power allocation coefficients $P_1, \ldots, P_L$ are of order $\frac{1}{L}$ and satisfy $P_1 + \ldots + P_L = P$.

with the highest power, then the sections with the next highest power etc. Correctly recovering a subset of the indices, say allows the decoder to cancel their contribution from y, thereby making the decoding of the remaining sections easier.

In the next section, we discuss an adaptive 'hard-decision' decoder based on the successive cancellation idea above. In Sections 3.3 and 3.4, we discuss two 'soft-decision' versions of the iterative decoder. All three decoders are asymptotically capacity-achieving, but the soft-decision decoders have better finite length error performance. In the next chapter, we will discuss how to design alternative power allocations to optimize finite length error performance.

We emphasize that the decoders above do not pre-specify an order in which the sections are decoded. Rather, the power allocation makes it likely that sections with higher power are decoded before those with lower power. This is similar in spirit to how algorithms such as Orthogonal Matching Pursuit for recovering sparse vectors can be significantly more powerful when the magnitudes of the non-zero coefficients have a decaying profile [58].

3.1 Adaptive successive hard-decision decoding

For our theoretical results we will use the following exponentially decaying allocation, with the power in section ℓ proportional to $e^{-2\mathcal{C}\ell/L}$:

$$P_\ell = P \cdot \frac{e^{2\mathcal{C}/L} - 1}{1 - e^{-2\mathcal{C}}} \cdot e^{-2\mathcal{C}\ell/L}, \quad \ell \in [L]. \tag{3.2}$$

Recalling that $\mathcal{C} = \frac{1}{2}\log(1 + \mathsf{snr})$, we note that $1 - e^{-2\mathcal{C}} = \mathsf{snr}/(1 + \mathsf{snr})$.

This power allocation is motivated by thinking of the L sections of the SPARC as corresponding to L users of a Gaussian multiple-access channel (MAC) with total power constraint P. Indeed, consider the equal-rate point on the capacity region of a L-user Gaussian MAC where each user gets rate $\mathcal{C}/L$. It is well-known [32, 37] that this rate point can be achieved via successive cancellation decoding, where user 1 is first decoded, then user 2 is decoded after subtracting the codeword of user 1, and so on. For this successive cancellation scheme, the power allocation for the L users is determined by the following set of equations:

$$\frac{1}{2}\log\left(1 + \frac{P_\ell}{\sigma^2 + P_{\ell+1}\ldots + P_L}\right) = \frac{\mathcal{C}}{L}, \quad \ell \in [L]. \tag{3.3}$$

Sequentially solving the set of equations in (3.3), starting from $\ell = L$, yields the exponentially decaying power allocation in (3.2).

Continuing the analogy with an L-user MAC, we ask: can the above successive cancellation scheme be used for SPARC decoding to achieve rates close to $\mathcal{C}$? Unfortunately, successive cancellation performs poorly for SPARC decoding. This is because L, the number of sections ('users') in the codebook grows with n. Indeed, for the choice $M = L^{\mathsf{a}}$, L grows as $n/\log n$, while M, the number of codewords per user, only grows polynomially in n. An error in decoding one section affects the decoding of future sections, leading to a large number of section errors after L steps. We note that

in the standard MAC set-up, the number of users L remains constant as the code length n grows; hence the rate per user is also of constant order.

The first feasible SPARC decoder, proposed in [65], controls the accumulation of section errors using *adaptive* successive decoding. The idea is to *not* pre-specify the order in which sections are decoded, but to look across all the undecoded sections in each step, and adaptively decode columns which have a large inner product with the residual. The adaptive successive decoding algorithm proceeds as follows.

Given $y = A\beta + w$, start with estimate $\beta^0 = 0$.

Initial step $[t = 1]$

1. Compute the inner product of $\sqrt{n}\, y/\|y\|$ with each column of A.

2. Pick the columns corresponding to inner products that cross a threshold $\sqrt{2 \log M} + a$ to form β^1, for a fixed constant $a > 0$.

3. Form the initial fit as weighted sum of columns: $\mathsf{Fit}^1 = A\beta^1$.

Iterate $[\text{step } t + 1, \; t \geq 1]$

1. Compute the normalized residual $\mathsf{Res}^t = \sqrt{n}\, (y - \mathsf{Fit}^t)/\|y - \mathsf{Fit}^t\|$.

2. Compute the inner product of Res^t with each *remaining* column of A.

3. Pick the columns that cross the threshold $\sqrt{2 \log M} + a$ to form β^{t+1}.

4. Compute the new fit $\mathsf{Fit}^{t+1} = A\beta^{t+1}$.

Stop if there are no additional inner products above threshold, or after $(\mathsf{snr}) \log M$ steps.

3.1.1 Intuition and analysis

First step The key observation is that in Step 1, the columns of A that are not sent (i.e., correspond to a zero entries in β) will produce normalized inner products whose joint distribution is close to i.i.d. $\mathcal{N}(0,1)$. On the other hand, the column that was sent in section ℓ will produce an inner product that is close to a standard normal plus a shift of size $\sqrt{nP_\ell/(P+\sigma^2)}$. This is made precise in the following lemma.

Lemma 3.1. *[65, Lemma 3] For $j \in [ML]$, let A_j denote the jth column of A, and let $\mathcal{Z}_{1,j} = \sqrt{n} A_j^* y/\|y\|$. Then for $j \in$ section ℓ, $\ell \in [L]$ we have*

$$\mathcal{Z}_{1,j} \stackrel{d}{=} \sqrt{\frac{nP_\ell}{P+\sigma^2}} \frac{\chi_n}{\sqrt{n}} \mathbf{1}\{j \text{ sent}\} + N_{1,j}, \tag{3.4}$$

25

where $N_1 = (N_{1,j} : 1 \leq j \leq ML)$ *is multivariate normal with zero mean and covariance matrix* $I - \frac{\beta\beta^*}{n(P+\sigma^2)}$. *Furthermore,* $\chi_n^2 = \|y\|^2/(P+\sigma^2)$ *is a Chi-square n random variable that is independent of* N_1.

Proof. Recall from (3.1) that $y = \sqrt{nP_1}A_{i_1} + \ldots + \sqrt{nP_L}A_{i_L} + w$, where $i_1, \ldots, i_L$ denote the indices of the sent terms. Also recall that $w \sim \mathcal{N}(0, \sigma^2 I)$ and $A_j \sim \mathcal{N}(0, \frac{1}{n}I)$ are i.i.d. for $1 \leq j \leq ML$. Using this we find that the conditional distribution of A_j given y, for j in section ℓ, is:

$$A_j \mid y \sim \begin{cases} \mathcal{N}(0, \frac{1}{n}I) & \text{if } j \neq i_\ell, \\ \mathcal{N}\left(y \frac{\sqrt{nP_\ell}}{n(P+\sigma^2)}, \frac{1}{n}(1 - \frac{P_\ell}{P+\sigma^2})I\right) & \text{if } j = i_\ell. \end{cases} \tag{3.5}$$

Hence the conditional distribution of A_j given y may be expressed as

$$A_j = \frac{1}{\sqrt{n}}\left(\frac{\beta_j}{P+\sigma^2}\frac{y}{\sqrt{n}} + U_j\right) \tag{3.6}$$

where $U_j \sim \mathcal{N}(0, (1 - \frac{\beta_j^2}{n(P+\sigma^2)}))$ is independent of y. Moreover, for a given row index i, since $\mathbb{E}[A_{j,i}A_{k,i}] = \frac{1}{n}\mathbf{1}\{j = k\}$, we have $\mathbb{E}[U_{j,i}U_{k,i}] = \frac{-\beta_j\beta_k}{n(P+\sigma^2)}$ for $j \neq k$. Therefore, for any $i \in [n]$ the random vector $(U_{1,i}, \ldots, U_{ML,i})$ has distribution $\mathcal{N}(0, (1 - \frac{\beta\beta^*}{n(P+\sigma^2)})I)$.

From (3.6) we have

$$\mathcal{Z}_{1,j} = \sqrt{n}A_j^*\frac{y}{\|y\|} = \frac{\beta_j}{\sqrt{P+\sigma^2}}\frac{\|y\|}{\sqrt{n(P+\sigma^2)}} + \frac{U_j^*y}{\|y\|}. \tag{3.7}$$

Letting $N_{1,j} = \frac{U_j^*y}{\|y\|}$ and $N_1 = (N_{1,j} : 1 \leq j \leq ML)$, to complete the proof we need to show that N_1 is a multivariate normal that is independent of y with covariance matrix $I - \frac{\beta\beta^*}{n(P+\sigma^2)}$. Indeed, conditioning on any (non-zero) realization of y it is seen that N_1 is a $\mathcal{N}(0, (1 - \frac{\beta\beta^*}{n(P+\sigma^2)})I)$ random vector. This completes the proof. $\square$

In Lemma 3.1, since $\chi_n/\sqrt{n}$ is close to 1 for large n, the shift in the inner product corresponding to the sent term in section in ℓ is

$$\sqrt{\frac{nP_\ell}{P+\sigma^2}} = \sqrt{\frac{LP_\ell}{R(P+\sigma^2)}\log M} \stackrel{(a)}{=} \sqrt{\frac{\mathcal{C}}{R}(1 + O(\frac{1}{L}))e^{-2\mathcal{C}\ell/L}}\sqrt{2\log M}, \tag{3.8}$$

where (a) is obtained using the exponential power allocation in (3.2), and the fact that $e^{2\mathcal{C}/L} - 1 = \frac{2\mathcal{C}}{L}(1 + O(\frac{1}{L}))$. Since $R < \mathcal{C}$ we observe from (3.8) that the shift will be larger than $\sqrt{2\log M}$ for $1 \leq \ell \leq \ell_0$, where ℓ_0 is determined by $\mathcal{C}/R$.

On the other hand, for any column j that is *not* sent in section ℓ, the shift is zero, and the test statistic $\mathcal{Z}_{1,j}$ normal. Recalling that each section has M columns, we note that the maximum of M standard normals concentrates near $\sqrt{2\log M}$ for large M [56]. Therefore, if the constant a defining the threshold is chosen to be small compared to $\sqrt{2\log M}$, then the true columns in sections $1 \leq \ell \leq \ell_0$ are likely to have inner products that exceed the threshold $\sqrt{2\log M} + a$. On the hand, $a > 0$ ensures that the probability of inner product of a wrong column crossing the threshold is small. It is evident that the value of a determines the trade-off between the probabilities of false alarm and missed detection.

Subsequent steps Let dec_t denote the set of sections decoded up to the end of step t. Then the residual Res_t removes the contribution of the sections in dec_t from y. Assuming that no mistakes were made until step t, by analogy with the Step 1 analysis above we expect the shift for the sent term in (a yet to be decoded) section ℓ to be close to

$$\sqrt{\frac{nP_\ell}{\sigma^2 + P(1 - x_t)}},\tag{3.9}$$

where $x_t = \frac{1}{P}\sum_{k\in\mathsf{dec}_t} P_k$ is the fraction of power that has already been decoded. Thus as decoding successfully progresses, x_t increases with t, making the shift in (3.9) larger and facilitating the decoding of sections with lower power.

However, establishing a result analogous to Lemma 3.1 for $t > 1$ is challenging. This is because the dependence between the residual Res_t and the matrix A cannot be easily characterized. Indeed, recall that Res_t has been generated via decisions based on inner products with columns on A computed in previous steps.

To address this, Barron and Joseph consider a slightly modified version of the decoder, where at the end of each step t, we compute G_t, the part of Fit_t that is orthogonal to $y, \mathsf{Fit}_1, \ldots, \mathsf{Fit}_{t-1}$. That is, with $G_0 \triangleq y$, the collection

$$\frac{G_0}{\|G_0\|}, \frac{G_1}{\|G_1\|}, \ldots, \frac{G_t}{\|G_t\|}$$

forms an orthonormal basis for $\mathsf{Fit}_1, \ldots, \mathsf{Fit}_t$. Then in step $(t + 1)$, instead of residual-based inner products, we compute the following test statistic for each column j in an undecoded section:

$$\mathcal{Z}_{t,j}^{\mathsf{comb}} = (A_j)^* \left[\lambda_0 \frac{G_0}{\|G_0\|} + \ldots + \lambda_t \frac{G_t}{\|G_t\|} \right]\tag{3.10}$$

where $\lambda_0, \ldots, \lambda_t$ are deterministic positive constants such that $\sum_{k=0}^{t} \lambda_k^2 = 1$. For an appropriate choice of λ_k's, the test statistic in (3.10) closely mimics the residual based statistic. Essentially, λ_k is chosen to be a deterministic proxy for the inner product

$$\frac{(\mathsf{Res}_t)^* G_k}{\|\mathsf{Res}_t\| \|G_k\|}.$$

With this choice, the test statistic $\mathcal{Z}_{t,j}^{\mathsf{comb}}$ can be shown to have a distributional representation that is approximately a shifted normal. That is,

$$\mathcal{Z}_{t,j}^{\mathsf{comb}} \stackrel{(a)}{\approx} \sqrt{\frac{nP_\ell}{\sigma^2 + P(1 - x_t)}} \mathbf{1}\{j \text{ sent}\} + N_{t,j},\tag{3.11}$$

where $N_{t,j}$ is normal zero-mean random variable with variance near 1. The parameter x_t quantifies the expected success rate, and can be interpreted as the expected fraction of power in the sections decoded by the end of iteration t. It can be recursively computed as follows, starting from $x_0 = 0$. With $\tau = \sqrt{2\log M} + a$ denoting the threshold used in each step and Φ denoting the standard

Figure 3.2: Evolution of $(1 - x_t)$ with iteration t. The SPARC parameters are $M = 512, L = 1024, \mathsf{snr} = 15, R = 0.8\mathcal{C}, P_\ell \propto e^{-2\mathcal{C}\ell/L}$ with $\mathcal{C}$ in nats. Curves are shown for three different values of the threshold $\tau = \sqrt{2\log M} + a$, with $a = 1, 0.8, 0$. The dashed curve shows the evolution of $(1 - x_t)$ for the soft-decison decoder discussed in the next section.

normal distribution function, we have

$$x_{t+1} = \sum_{\ell=1}^{L} \frac{P_\ell}{P} \, \Phi\left(\sqrt{\frac{nP_\ell}{\sigma^2 + P(1 - x_t)}} - \tau \right) \tag{3.12}$$

$$= \sum_{\ell=1}^{L} \frac{P_\ell}{P} \, \Phi\left(\sqrt{2\log M} \left(\sqrt{\frac{\mathcal{C}}{R} \cdot \frac{\sigma^2 + P}{\sigma^2 + P(1 - x_t)} e^{-2\mathcal{C}\ell/L}} - 1 \right) - a + O(\tfrac{1}{L}) \right) \tag{3.13}$$

where (3.13) is obtained using the expression for $\sqrt{nP_\ell}$ from (3.8).

Figure 3.2 shows the progression of $(1 - x_t)$ with t, for three different values of the threshold $\sqrt{2\log M} + a$, with $a = 1, 0.8, 0$. The parameter $(1 - x_t)$ quantifies the expected fraction of power in the undecoded sections after iteration t. Observe from (3.12) that a smaller value of the threshold τ results in a smaller value of $(1 - x_t)$; this is illustrated by the curves shown in Fig. 3.2. However, the recursive formula (3.12) is an idealized prediction: it gives the expected fraction of power in the decoded sections at the end of each iteration t, assuming that there are no *false alarms*, i.e., no sections have been incorrectly decoded. For finite block lengths the value of $a > 0$ in the threshold $\tau = \sqrt{2\log M} + a$ plays a crucial role in determining the false alarm rate. The larger the value of a, the lower the probability of false alarms.

A rigorous statement specifying the distributional representation of $\mathcal{Z}_{t,j}^{\mathsf{comb}}$, taking into account the false alarm rate, is given in [65, Lemma 4]. This representation leads to the following performance guarantee for the decoder, which in essence states that rates up to

$$\mathcal{C}^* := \frac{\mathcal{C}}{1 + \delta_M} \tag{3.14}$$

can be achieved with $O(\delta_M)$ fraction of section errors, where

$$\delta_M := \frac{1}{\sqrt{\pi \log M}}. \tag{3.15}$$

28

Theorem 3.1. *[65, Theorem 2] Let the rate $R < \mathcal{C}^*$ be expressed in the form*

$$\frac{\mathcal{C}^*}{1 + \frac{\kappa}{\log M}} \tag{3.16}$$

with $\kappa > 0$. Then, with the exponentially decaying power allocation in (3.2) the adaptive successive decoder has section error rate less than

$$\delta_{err} := \frac{1}{2\mathcal{C}\sqrt{\pi \log M}} + \frac{3\kappa + 5}{8\mathcal{C}\log M} \tag{3.17}$$

with probability at least $1 - P_e$, where

$$P_e = \kappa_{1,M} e^{-\kappa_2 L \min\{\kappa_3 \Delta^2, \ \kappa_4 \Delta\}}. \tag{3.18}$$

In (3.18), $\Delta = \frac{\mathcal{C}^* - R}{\mathcal{C}^*}$, $\kappa_{1,M}$ is a polynomial in M, and κ_2, κ_3 and κ_4 are constants that depend on snr.

Remark 3.1. *As in Proposition 2.2, we can concatenate the SPARC with an outer Reed-Solomon code of rate $(1 - 2\delta_{err})$ to guarantee that the message error probability is bounded by P_e in (3.18). Thus Theorem 3.1 tells us that the adaptive successive decoder can achieve rates of the order of $1/\sqrt{\log M}$ below capacity.*

Choosing $L = M^a$, we have M of order $(n/\log n)^a$, and hence the minimum gap from capacity is of order $1/\sqrt{\log n}$. This gap is much larger than that of the optimal decoder, which can achieve rates up to order $1/n^\alpha$ below capacity with error probability decaying exponentially in $n^{1-2\alpha}$, for any $\alpha \in (0, \frac{1}{2})$ (see Remark 2.3).

Remark 3.2. *It is shown in [64, Sec. 4.18] that the gap from capacity can be improved to $O(\log \log M/\log M)$ using a power allocation that is slightly modified from the one in (3.2). The power P_ℓ is now chosen proportional to*

$$\max\{e^{-\frac{2\mathcal{C}\ell}{L}}, \ e^{-2\mathcal{C}}(1 + \frac{c}{\sqrt{2\log M}})\},$$

for a suitably chosen constant c. This allocation slightly boosts the power for sections ℓ close to L. This helps ensure that, even towards the end of the algorithm, there will be sections for which the true terms are expected to have inner product above threshold.

3.2 Iterative soft-decision decoding

Theorem 3.1 shows that the adaptive successive hard-thresholding decoder is asymptotically capacity-achieving. However, the empirical section error rate at practically feasible code lengths is rather high for rates near capacity. We now discuss two soft-decision decoders, the adaptive successive soft-decision decoder and the approximate message passing (AMP) decoder, which have better error performance at finite code lengths. Instead of making hard decisions about which columns to decode in each step, the soft-decision decoders generate iteratively refined estimates of the message vector in each step. Both soft-decision decoders share a few key underlying principles. We

first discuss these principles in this section. The specifics of the two decoding algorithms are then described in the next two sections.

The decoder starts with $\beta^0 = 0$ (the all-zero vector of length ML), and generates an updated estimate of the message vector in each step; these estimates are denoted by $\beta^1, \beta^2, \ldots$. The key idea in soft-decision decoding is to form the new estimate in each step by updating the posterior probabilities of each entry of β being the true non-zero in its section. This is done as follows.

At the end of each step t, the decoder produces a test statistic $\mathsf{stat}^t \in \mathbb{R}^{ML}$ that has the form

$$\mathsf{stat}^t \approx \beta + \tau_t Z, \tag{3.19}$$

where Z is a standard normal random vector independent of β. That is, stat^t is approximately distributed as the true message vector plus an independent standard Gaussian vector with known variance τ_t^2. The test statistic stat^t is produced based on y, A and the previous estimates $\beta^1, \ldots, \beta^t$. The details of how stat^t is produced to ensure that (3.19) holds depend on the type of soft-decision decoder used. These details are described in Sections 3.3 and 3.4.

In step $(t + 1)$, the decoder generates an updated estimate β^{t+1} based on stat^t. Assuming that the distributional property in (3.19) exactly holds at the end of step t, the Bayes-optimal estimate for β that minimizes the expected squared error in the next step $(t + 1)$ is

$$\beta^{t+1} = \eta^t(\mathsf{stat}^t) := \mathbb{E}[\beta \mid \beta + \tau_t Z = \mathsf{stat}^t]. \tag{3.20}$$

The conditional expectation above can be computed as follows using the known prior on β in which the location of the non-zero within section is uniformly random. For $\mathsf{stat}^t = s = (s_1, \ldots, s_{ML})$ and index $i \in \sec(\ell)$, $\ell \in [L]$ we have

$$
\begin{aligned}
\eta_i^t(\mathsf{stat}^t = s) &= \mathbb{E}[\beta_i \mid \beta + \tau_t Z = s] = \mathbb{E}[\beta_i \mid \{\beta_j + \tau_t Z_j = s_j\}_{j \in \sec(\ell)}] \\
&= \sqrt{nP_\ell}\, P(\beta_i = \sqrt{nP_\ell} \mid \{\beta_j + \tau_t Z_j = s_j\}_{j \in \sec(\ell)}) \\
&= \frac{\sqrt{nP_\ell}\, f(\{s_j\}_{j \in \sec(\ell)} \mid \beta_i = \sqrt{nP_\ell})\, P(\beta_i = \sqrt{nP_\ell})}{\sum_{k \in \sec(\ell)} f(\{s_j\}_{j \in \sec(\ell)} \mid \beta_k = \sqrt{nP_\ell})\, P(\beta_k = \sqrt{nP_\ell})}
\end{aligned}
\tag{3.21}
$$

where we have used Bayes' theorem with $f(\cdot \mid \beta_k = \sqrt{nP_\ell})$ denoting the joint density of $\{\beta_j + \tau_t Z_j\}_{j \in \sec(\ell)}$ conditioned on β_k being the non-zero entry in section ℓ. Since β and Z are independent with Z having i.i.d. $\mathcal{N}(0, 1)$ entries, for each $k \in \sec(\ell)$ we have

$$
\begin{aligned}
&f(\{\beta_j + \tau_t Z_j = s_j\}_{j \in \sec(\ell)} \mid \beta_k = \sqrt{nP_\ell}) \\
&\propto e^{-(s_k - \sqrt{nP_\ell})^2 / 2\tau_t^2} \prod_{j \in \sec(\ell), j \neq k} e^{-s_j^2 / 2\tau_t^2}.
\end{aligned}
\tag{3.22}
$$

Using (3.22) in (3.21), together with the fact that $P(\beta_k = \sqrt{nP_\ell}) = \frac{1}{M}$ for each $k \in \sec(\ell)$, we obtain

$$\eta_i^t(\mathsf{stat}^t = s) = \mathbb{E}[\beta_i \mid \beta + \tau_t Z = s] = \sqrt{nP_\ell} \frac{e^{s_i \sqrt{nP_\ell} / \tau_t^2}}{\sum_{j \in \sec(\ell)} e^{s_j \sqrt{nP_\ell} / \tau_t^2}}. \tag{3.23}$$

3.2.1 State evolution

To compute β^{t+1} using (3.23) requires the parameter τ_t^2, which is the variance of the noise in the desired distributional representation $\mathsf{stat}^t = \beta + \tau_t Z$. This noise variance has two components: one is the channel noise variance σ^2, and the other is the mean-squared estimation error $\frac{1}{n}\mathbb{E}\|\beta - \beta^t\|^2$.

Starting with $\tau_0^2 = \sigma^2 + P$, we recursively compute τ_{t+1}^2 for $t \geq 0$ as follows:

$$\tau_{t+1}^2 = \sigma^2 + \frac{1}{n}\mathbb{E}\|\beta - \mathbb{E}[\beta|\beta + \tau_t Z]\|^2 = \sigma^2 + \frac{1}{n}\mathbb{E}\|\beta - \eta_t(\beta + \tau_t Z)]\|^2, \tag{3.24}$$

where the expectation on the right is over β and the independent standard normal vector Z. The recursion (3.24) to generate τ_{t+1}^2 from τ_t^2 can be written as

$$\tau_{t+1}^2 = \sigma^2 + P(1 - x_{t+1}) \tag{3.25}$$

where $x_{t+1} = x(\tau_t)$, with

$$x(\tau) := \sum_{\ell=1}^{L} \frac{P_\ell}{P} \mathbb{E}\left[\frac{\exp\left\{\frac{\sqrt{nP_\ell}}{\tau}\left(U_1^\ell + \frac{\sqrt{nP_\ell}}{\tau}\right)\right\}}{\exp\left\{\frac{\sqrt{nP_\ell}}{\tau}\left(U_1^\ell + \frac{\sqrt{nP_\ell}}{\tau}\right)\right\} + \sum_{j=2}^{M}\exp\left\{\frac{\sqrt{nP_\ell}}{\tau}U_j^\ell\right\}}\right]. \tag{3.26}$$

In (3.26), $\{U_j^\ell\}$ are i.i.d. $\mathcal{N}(0,1)$ random variables for $j \in [M]$ and $\ell \in [L]$. For consistency, we define $x_0 = 0$.

The equivalence between the recursions in (3.24) and (3.25) is established by the following proposition.

Proposition 3.2. *[95] Under the assumption that $\mathsf{stat}^t = \beta + \tau_t U$, where $U \in \mathbb{R}^{ML}$ is standard normal and independent of β, the quantity $x_{t+1} = x(\tau_t)$ satisfies*

$$x_{t+1} = \frac{1}{nP}\mathbb{E}[\beta^*\beta^{t+1}], \quad 1 - x_{t+1} = \frac{1}{nP}\mathbb{E}[\|\beta - \beta^{t+1}\|^2], \tag{3.27}$$

and consequently, (3.24) and (3.25) are equivalent.

Proof. For convenience of notation, we label the ML components of the standard normal vector U as $\{U_j^\ell\}_{j\in[M],\ell\in[L]}$. For any ℓ, U^ℓ denotes the length M vector $\{U_j^\ell\}_{j\in[M]}$. We have

$$\frac{1}{nP}\mathbb{E}[\beta^*\beta^{t+1}] = \frac{1}{nP}\mathbb{E}[\beta^* \eta^t(\beta + \tau_t U)]$$

$$\stackrel{(a)}{=} \frac{1}{nP}\sum_{\ell=1}^{L}\mathbb{E}[\sqrt{nP_\ell}\, \eta_{\mathsf{sent}(\ell)}^t(\beta_\ell + \tau_t U^\ell)]$$

$$\stackrel{(b)}{=} \frac{1}{nP}\sum_{\ell=1}^{L}\mathbb{E}\left[\sqrt{nP_\ell}\,\frac{\sqrt{nP_\ell}\cdot e^{\sqrt{nP_\ell}(\sqrt{nP_\ell}+\tau_t U_1^\ell)/\tau_t^2}}{e^{\sqrt{nP_\ell}(\sqrt{nP_\ell}+\tau_t U_1^\ell)/\tau_t^2} + \sum_{j=2}^{M}e^{\sqrt{nP_\ell}\tau_t U_j^\ell/\tau_t^2}}\right] \tag{3.28}$$

$$= \sum_{\ell=1}^{L}\frac{P_\ell}{P}\mathbb{E}\left[\frac{e^{\frac{\sqrt{nP_\ell}}{\tau_t}(U_1^\ell+\frac{\sqrt{nP_\ell}}{\tau_t})}}{e^{\frac{\sqrt{nP_\ell}}{\tau_t}(U_1^\ell+\frac{\sqrt{nP_\ell}}{\tau_t})} + \sum_{j=2}^{M}e^{\frac{\sqrt{nP_\ell}}{\tau_t}U_j^\ell}}\right] = x_{t+1}.$$

31

In (a) above, the index of the non-zero term in section ℓ is denoted by $\mathsf{sent}(\ell)$. Step (b) is obtained by assuming that $\mathsf{sent}(\ell)$ is the first entry in section ℓ — this assumption is valid because the prior on β is uniform over $\mathcal{B}_{M,L}(P_1, \ldots, P_L)$.

Next consider
$$\frac{1}{nP}\mathbb{E}[\|\beta - \beta^{t+1}\|^2] = 1 + \frac{\mathbb{E}[\|\beta^{t+1}\|^2] - 2\mathbb{E}[\beta^* \beta^{t+1}]}{nP}. \tag{3.29}$$

Under the assumption that $\mathsf{stat}^t = \beta + \tau_t Z$, recall from Section 3.2 that β^{t+1} can be expressed as $\beta^{t+1} = \mathbb{E}[\beta \mid \mathsf{stat}^t]$. We therefore have

$$\mathbb{E}[\|\beta^{t+1}\|^2] = \mathbb{E}[\|\mathbb{E}[\beta|\mathsf{stat}^t]\|^2] = \mathbb{E}[(\mathbb{E}[\beta|\mathsf{stat}^t] - \beta + \beta)^* \mathbb{E}[\beta|\mathsf{stat}^t]]$$
$$\stackrel{(a)}{=} \mathbb{E}[\beta^* \mathbb{E}[\beta|\mathsf{stat}^t]] = \mathbb{E}[\beta^* \beta^{t+1}], \tag{3.30}$$

where step (a) follows because $\mathbb{E}[(\mathbb{E}[\beta|\mathsf{stat}^t] - \beta)^* \mathbb{E}[\beta|\mathsf{stat}^t]] = 0$ due to the orthogonality principle. Substituting (3.30) in (3.29) and using (3.28) yields

$$\frac{1}{nP}\mathbb{E}[\|\beta - \beta^{t+1}\|^2] = 1 - \frac{\mathbb{E}[\beta^* \beta^{t+1}]}{nP} = 1 - x_{t+1}.$$

$\square$

The parameter x_t can be interpreted as the power-weighted fraction of sections correctly decodable after step t: starting from $x_0 = 0$. The recursion defined by (3.26) and (3.25) to compute the parameters $(x_t, \tau_t^2)_{t=0,1,\ldots}$ is called *state evolution*. This terminology is due to the similarity with density evolution, the recursion used to predict the performance of LDPC codes [93].

Figure 3.2 on page 28 shows the progression of $(1 - x_t)$ for soft-decision decoding in dashed lines, alongside the solid lines for hard-decision decoding. For the soft-decision case, x_t is recursively computed using the state evolution recursion in (3.26) and (3.25). As we do not make hard decisions on decoded columns until the end, there is no false alarm rate to be controlled in each iteration. If the iterative soft decision decoder is run for T steps, we wish to ensure that x_T is as close to one as possible, implying that the expected squared error $\frac{1}{n}\mathbb{E}\|\beta - \beta^T\|^2 \approx 0$ under the distributional assumption for stat^t.

Figure 3.3 shows the progression of the MSE $\frac{1}{n}\|\beta - \beta^t\|^2$ for 200 trials of the AMP decoder (green curves); it is seen that the average is closely tracked by $(1 - x_t)$ (black curve). The theoretical analysis of the soft-decision decoders discussed in the next two sections shows that the decoding performance of the soft-decision decoders in each step t is closely tracked by the parameter x_t as the SPARC parameters (L, M, n) grow large.

The following lemma specifies the state evolution recursion in the large system limit, i.e., as $L, M, n \to \infty$ such that $L \log M = nR$. We denote this limit by $\lim$.

Lemma 3.3. *[95, Lemma 1] For any power allocation $\{P_\ell\}_{\ell=1,\ldots,L}$ that is non-increasing with ℓ, we have*
$$\bar{x}(\tau) := \lim x(\tau) = \lim \sum_{\ell=1}^{\lfloor \xi^*(\tau)L \rfloor} \frac{P_\ell}{P}, \tag{3.31}$$

32

Figure 3.3: Comparison of state evolution predictions with AMP performance. The SPARC parameters are $M = 512, L = 1024, \mathsf{snr} = 15, R = 0.7\mathcal{C}, P_\ell \propto e^{-2\mathcal{C}\ell/L}$ with $\mathcal{C}$ in nats. The average of the 200 trials (green curves) is the dashed red curve, which is almost indistinguishable from the state evolution prediction (black curve).

where $\xi^*(\tau)$ is the supremum of all $\xi \in (0, 1]$ that satisfy

$$\lim LP_{\lfloor \xi L \rfloor} > 2R\tau^2.$$

If $\lim LP_{\lfloor \xi L \rfloor} \leq 2R\tau^2$ for all $\xi > 0$, then $\bar{x}(\tau) = 0$. (The rate R is measured in nats.)

Proof. In Sec. 3.6.1. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\square$

Recalling that $x_{t+1} = x(\tau_t)$ is the expected power-weighted fraction of correctly decoded sections after step $(t + 1)$, for any power allocation $\{P_\ell\}$, Lemma 3.3 can be interpreted as follows: in the large system limit, sections ℓ such that $\ell \leq \lfloor \xi^*(\bar{\tau}_t)L \rfloor$ will be correctly decodable in step $(t + 1)$, i.e., the soft-decision decoder will assign most of the posterior probability mass to the correct term. Conversely all sections whose power falls below the threshold will not be decodable in this step.

For the exponentially decaying power allocation in (3.2), we have for $\xi \in (0, 1]$:

$$\lim LP_{\lfloor \xi L \rfloor} = \sigma^2(1 + \mathsf{snr})^{1-\xi} \ln(1 + \mathsf{snr}). \tag{3.32}$$

Using this in Lemma 3.3 yields the following result.

Lemma 3.4. *[95, Lemma 2] For the power allocation $\{P_\ell\}$ given in (3.2), we have for $t = 0, 1, \ldots$:*

$$\bar{x}_t := \lim x_t = \frac{(1 + \mathsf{snr}) - (1 + \mathsf{snr})^{1-\xi_{t-1}}}{\mathsf{snr}}, \tag{3.33}$$

$$\bar{\tau}_t^2 := \lim \tau_t^2 = \sigma^2 + P(1 - \bar{x}_t) = \sigma^2 (1 + \mathsf{snr})^{1-\xi_{t-1}} \tag{3.34}$$

where $\xi_{-1} = 0$, and for $t \geq 0$,

$$\xi_t = \min \left\{ \left(\frac{1}{2\mathcal{C}} \log \left(\frac{\mathcal{C}}{R} \right) + \xi_{t-1} \right), \ 1 \right\}. \tag{3.35}$$

33

Proof. The result is obtained by applying Lemma 3.3 with the exponential power allocation, and using induction on t. $\qquad\square$

A direct consequence of (3.33) and (3.35) is that $\bar{x}_t$ strictly increases with t until it reaches one, and the number of steps T^* until $\bar{x}_{T^*} = 1$ is $T^* = \left\lceil \frac{2\mathcal{C}}{\log(\mathcal{C}/R)} \right\rceil$.

The constants $\{\xi_t\}_{t \geq 0}$ have a nice interpretation in the large system limit: at the end of step $t+1$, the first ξ_t fraction of sections in β^{t+1} will be correctly decodable with high probability. The other $(1 - \xi_t)$ fraction of sections *will not* be correctly decodable from β^{t+1} as the power allocated to these sections is not large enough. An additional $\frac{1}{2\mathcal{C}} \log \left(\frac{\mathcal{C}}{R} \right)$ fraction of sections become correctly decodable in each step until step T^*, when all the sections are correctly decodable with high probability.

The discussion in this section — starting from the way the estimates $(\beta^t)_{t \geq 1}$ are generated, up to the interpretation of the state evolution parameters $(x_t, \tau_t^2)_{t \geq 0}$ — has been based on the assumption that the decoder has available test statistics of the form $\mathsf{stat}^t = \beta + \tau_t Z$ at the end of each iteration. In the next two sections, we will describe two decoders which produce stat^t of approximately this form when the SPARC parameters (L, M, n) are sufficiently large.

3.3 Adaptive successive soft-decision decoder

The soft-decision decoder proposed by Barron and Cho [17, 27, 26] computes the test statistic stat^t at the end of step t as a function of $(A, y, \mathsf{Fit}_1, \ldots, \mathsf{Fit}_t)$, where we recall $\mathsf{Fit}_t = A\beta^t$. As in hard decision decoding (see p. 27), starting with $G_0 \triangleq y$, let G_t be the part of Fit_t that is orthogonal to $G_0, \ldots, G_{t-1}$. Then the collection

$$\frac{G_0}{\|G_0\|}, \frac{G_1}{\|G_1\|}, \ldots, \frac{G_t}{\|G_t\|}$$

forms an orthonormal basis for $y, \mathsf{Fit}_1, \ldots, \mathsf{Fit}_t$. Also define, for $t \geq 0$:

$$\mathcal{Z}_t = \sqrt{n} \frac{A^* G_t}{\|G_t\|} \tag{3.36}$$

We compute a linear combination of $\mathcal{Z}_0, \ldots, \mathcal{Z}_t$ given by

$$\mathcal{Z}_t^{\mathsf{comb}} = \lambda_0 \mathcal{Z}_0 + \ldots + \lambda_t \mathcal{Z}_t, \tag{3.37}$$

where $\lambda_0, \ldots, \lambda_t$ are coefficients chosen such that $\sum_{k=0}^{t} \lambda_k^2 = 1$. (These coefficients may depend on A and y.) The adaptive successive soft-decision decoder then computes the statistic $\mathsf{stat}^t = \tau_t \mathcal{Z}_t^{\mathsf{comb}} + \beta^t$, where τ_t is the state evolution parameter defined in (3.25) and β^t is the estimate at the end of step t. The new estimate is generated as $\beta^{t+1} = \eta_t(\mathsf{stat}^t)$, where η_t is defined in (3.23). The algorithm is summarized in Fig. 3.4.

The key question is: how do we choose coefficients $\underline{\lambda}_t = (\lambda_0, \ldots, \lambda_t)$ such that stat^t has the desired representation $\mathsf{stat}^t \approx \beta + \tau_t Z$. To answer this, we use the following lemma which specifies the

Step 0: Initialize $\beta^0 = 0$ and $G_0 = y$.

Step $t + 1$, for $0 \leq t \leq (T-1)$:

1. Compute $\mathsf{Fit}_t = A\beta^t$

2. If $t \geq 1$, compute G_t, the orthogonal projection of Fit_t onto the space orthogonal to $G_0, \ldots, G_{t-1}$.

3. Compute $\mathcal{Z}_t = \sqrt{n}\, A^* G_t / \|G_t\|$, and

$$\mathcal{Z}_t^{\mathsf{comb}} = \lambda_0 \mathcal{Z}_0 + \ldots + \lambda_t \mathcal{Z}_t,$$

where $(\lambda_0, \ldots, \lambda_t)$ are given by (3.51).

4. Compute $\mathsf{stat}^t = \tau_t \mathcal{Z}_t^{\mathsf{comb}} + \beta^t$ where τ_t is given by (3.25).

5. Generate the updated estimate $\beta^{t+1} = \eta_t(\mathsf{stat}^t)$, where η_t is defined in (3.23).

The number of iterations T is determined using the state evolution recursion, as discussed on p. 38.

Figure 3.4: Adaptive successive soft-decision decoder with deterministic coefficients of combination.

conditional distribution of the components $\mathcal{Z}_t$ defined in (3.36). We need some definitions before stating the result.

Let $b_{0,e}, b_{1,e}, \ldots, b_{t,e} \in \mathbb{R}^{ML+1}$ be the successive orthonormal components of the length of the *extended* vectors

$$\beta_e := \begin{bmatrix} \beta \\ \sqrt{n}\,\sigma \end{bmatrix}, \quad \beta_e^1 := \begin{bmatrix} \beta^1 \\ 0 \end{bmatrix}, \quad \ldots, \quad \beta_e^t := \begin{bmatrix} \beta^t \\ 0 \end{bmatrix}. \tag{3.38}$$

Let $b_0, \ldots, b_t \in \mathbb{R}^{ML}$ be the vectors formed from the upper ML coordinates of $b_{0,e}, \ldots, b_{t,e}$. Let $\Sigma_{t,e} = \mathsf{I} - b_{0,e} b_{0,e}^* - b_{1,e} b_{1,e}^* \ldots - b_{t,e} b_{t,e}^*$ denote the $(ML+1) \times (ML+1)$ projection matrix onto the space orthogonal to the vectors in (3.38). The upper left $ML \times ML$ portion of this matrix is denoted Σ_t.

Lemma 3.5. *[17, Lemma 1] For $t \geq 0$, let*

$$\mathcal{F}_{t-1} = (\mathcal{Z}_0, \|G_0\|, \ldots, \mathcal{Z}_{t-1}, \|G_{t-1}\|),$$

with $\mathcal{F}_{-1}$ being the empty set. Then for $t \geq 0$, given $\mathcal{F}_{t-1}$, the conditional distribution $\mathbb{P}_{\mathcal{Z}_t | \mathcal{F}_{t-1}}$ of $\mathcal{Z}_t$ is determined by the representation

$$\mathcal{Z}_t = b_t \frac{\|G_t\|}{\varsigma_t} + Z_t, \tag{3.39}$$

where Z_t has conditional distribution $\mathcal{N}(0, \Sigma_t)$. Here, $\varsigma_0^2 = \sigma^2 + P$ and for $t \geq 1$ it is $\varsigma_t^2 = \hat{\beta}_t^* \varsigma_{t-1} \beta^t$. Moreover, $\|G_t\|^2 / \varsigma_t^2$ is distributed as a χ_{n-t}^2 random variable independent of Z_t and $\mathcal{F}_{t-1}$.

35

As number of iterations of the algorithm is small compared to n, $\frac{\|G_t\|}{\varsigma_t}$ is close to $\sqrt{n}$. The lemma tells us that for each t, $\mathcal{Z}$ is approximately equal to $\sqrt{n}b_t$ plus a standard normal vector. We use this property to choose coefficients $(\lambda_0, \ldots, \lambda_t)$ which lead to stat^t having the desired form.

Idealized coefficients. Consider the choice $\underline{\lambda}_t^{\mathsf{id}} = (\lambda_0, \ldots, \lambda_t)$ given by

$$\underline{\lambda}_t^{\mathsf{id}} = \frac{1}{c_t^{\mathsf{id}}} \left((\sqrt{n(P + \sigma^2)} - b_0^* \beta^t), -b_1^* \beta^t, \ldots, -b_t^* \beta^t \right), \tag{3.40}$$

where c_t^{id} is a normalizing constant to ensure that $\sum_k \lambda_k^2 = 1$. Since $b_0 = \beta / \sqrt{n(\sigma^2 + P)}$ and the decoder does not know β, this choice of coefficients cannot be used in practice. We call these idealized coefficients because understanding the test statistic produced by these will help us design good deterministic or observation-based coefficients.

Recalling that $b_{0,e}, \ldots, b_{t,e}$ form an orthonormal basis, the normalizing constant in (3.40) is computed as

$$
\begin{aligned}
(c_t^{\mathsf{id}})^2 &= (\sqrt{n(P + \sigma^2)} - b_0^* \beta^t)^2 + (-b_1^* \beta^t)^2 + \ldots + (-b_t^* \beta^t)^2 \\
&= n(\sigma^2 + P) + \|\beta^t\|^2 - 2\sqrt{P + \sigma^2} \frac{\beta^* \beta^t}{\sqrt{P + \sigma^2}} = n\sigma^2 + \|\beta - \beta^t\|^2,
\end{aligned}
\tag{3.41}
$$

where we have used the fact that $\|\beta\|^2 = nP$.

Let us now examine the distributional properties of the $\mathcal{Z}_t^{\mathsf{comb}}$ generated using these idealized coefficients via (3.37). Lemma 3.3 tells us that given $\mathcal{F}_{k-1}$, $\mathcal{Z}_k$ is closely approximated by $\sqrt{n}b_k + Z_k$ with Z_k standard normal, for $0 \leq k \leq t$. Therefore, with the idealized coefficients we obtain

$$
\begin{aligned}
\mathcal{Z}_t^{\mathsf{comb}} &= \lambda_0 \mathcal{Z}_0 + \ldots + \lambda_t \mathcal{Z}_t \\
&\overset{d}{\approx} \frac{\sqrt{n(P + \sigma^2)} \sqrt{n}b^0 - \left[(b_0^* \beta^t)\sqrt{n}b_0 + \ldots + (b_t^* \beta^t)\sqrt{n}b_t \right]}{c_t^{\mathsf{id}}} + Z \\
&= \frac{\sqrt{n}(\beta - \beta^t)}{c_t^{\mathsf{id}}} + Z = \frac{\beta - \beta^t}{\sqrt{\sigma^2 + \|\beta - \beta^t\|^2/n}} + Z,
\end{aligned}
\tag{3.42}
$$

where $Z \in \mathbb{R}^{ML}$ is standard normal. If we assume (via an induction hypothesis) that $\mathsf{stat}_{t-1} = \beta + \tau_{t-1} Z'$ for a standard normal vector $Z' \in \mathbb{R}^{ML}$, then Proposition 3.2 tells us that $\frac{1}{n}\mathbb{E}[\|\beta - \beta^t\|^2] = P(1 - x_t)$. Therefore, for large n, the term

$$\sigma^2 + \frac{\|\beta - \beta^t\|^2}{n} \approx \sigma^2 + P(1 - x_t) = \tau_t^2.$$

Hence the statistic $\mathsf{stat}^t = \tau_t \mathcal{Z}_t^{\mathsf{comb}} + \beta^t$ has the following approximate representation:

$$\mathsf{stat}^t = \tau_t \mathcal{Z}_t^{\mathsf{comb}} + \beta^t \approx \sqrt{\sigma^2 + \frac{\|\beta - \beta^t\|^2}{n}} \, \mathcal{Z}_t^{\mathsf{comb}} + \beta^t \overset{d}{=} \beta + \tau_t Z, \tag{3.43}$$

where the distributional representation follows from (3.42).

Deterministic coefficients. Since $b_0 = \beta/\sqrt{n(\sigma^2 + P)}$ is unknown, the idealized coefficients in (3.40) cannot be used to produce stat^t. We now specify a deterministic choice for λ_t which mimics the idealized coefficients using deterministic proxies for the inner products $b_0^* \beta^t, b_1^* \beta^t, \ldots, b_t^* \beta^t$. Recall that the vectors $b_{0,e}, \ldots, b_{1,e}$ are obtained by performing a successive orthonormalization on the extended vectors $(\beta_e, \beta_e^1, \ldots, \beta_e^t)$ defined in (3.38). We therefore have

$$
B := \begin{bmatrix} \beta_e & \beta_e^1, \ldots & \beta_e^t \end{bmatrix} = \begin{bmatrix} b_{0,e} & b_{1,e} & \ldots & b_{k,e} \end{bmatrix} \begin{bmatrix} b_{0,e}^* \beta_e & b_{0,e}^* \beta_e^1 & \ldots & b_{0,e}^* \beta_e^t \\ 0 & b_{1,e}^* \beta_e^1 & \ldots & b_{1,e}^* \beta_e^t \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & b_{t,e}^* \beta_e^t \end{bmatrix}. \tag{3.44}
$$

Noting that the last entry in each of $\beta_e^1, \ldots, \beta_e^t$ is zero, we observe that

$$
B^* B = \begin{bmatrix} \beta_e^* \beta_e & \beta^* \beta^1 & \ldots & \beta^* \beta^t \\ (\beta^1)^* \beta & (\beta^1)^* \beta^1 & \ldots & (\beta^1)^* \beta^t \\ \vdots & \vdots & \ddots & \vdots \\ (\beta^t)^* \beta & \ldots & \ldots & (\beta^t)^* \beta^t \end{bmatrix} = R^* R, \tag{3.45}
$$

where

$$
R = \begin{bmatrix} b_0^* \beta_e & b_0^* \beta^1 & \ldots & \boxed{b_0^* \beta^t} \\ 0 & b_1^* \beta^1 & \ldots & \boxed{b_1^* \beta^t} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \boxed{b_t^* \beta^t} \end{bmatrix}. \tag{3.46}
$$

The high-level idea in obtaining the deterministic coefficients is as follows. Observe that the entries in the last column of R (highlighted) are exactly those that are required to compute the idealized weights of combination in (3.40). These can be estimated by replacing each entry of the matrix in (3.45) with its idealized (deterministic) value, and then computing the Cholesky decomposition of this matrix. The last column of the resulting upper triangular matrix then provides a deterministic proxy for the highlighted terms in (3.46).

In detail, to obtain the deterministic coefficients we use the following result implied by Proposition 3.2: under the assumption (via an induction hypothesis) that $\mathsf{stat}^k = \beta + \tau_k Z_k'$ for $0 \leq k \leq (t-1)$, we have

$$
\frac{1}{n} \mathbb{E}[\beta^* \beta^k] = \frac{1}{n} \mathbb{E}[(\beta^m)^* \beta^k] = P x_k, \quad 1 \leq k \leq m \leq t. \tag{3.47}
$$

Therefore, replacing each entry of $B^* B/n$ by its expected value, we obtain the matrix

$$
\begin{bmatrix} \tau_0^2 & x_1 P & \ldots & x_t P \\ x_1 P & x_1 P & \ldots & x_1 P \\ \vdots & \vdots & \ddots & \vdots \\ x_t P & \ldots & \ldots & x_t P \end{bmatrix} = \widehat{R}^* \widehat{R}, \tag{3.48}
$$

where $\widehat{R}$ is the upper triangular matrix obtained via the Cholesky decomposition. This is found to

be

$$\widehat{R} = \begin{bmatrix} \tau_0 & \tau_0 - \tau_1^2\sqrt{\omega_0} & \cdots & \tau_0 - \tau_t^2\sqrt{\omega_0} \\ 0 & \tau_1^2\sqrt{\omega_1} & \cdots & \tau_t^2\sqrt{\omega_1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \tau_t^2\sqrt{\omega_t}, \end{bmatrix} \tag{3.49}$$

where

$$\omega_0 = \frac{1}{\tau_0^2}, \qquad \omega_k = \frac{1}{\tau_k^2} - \frac{1}{\tau_{k-1}^2}, \quad k \geq 1. \tag{3.50}$$

The last column of $\widehat{R}$ (highlighted) is a deterministic estimate for $(b_0^*\beta^t/\sqrt{n}, \ldots, b_t^*\beta^t/\sqrt{n})$. This is used to replace the idealized coefficients in (3.40), yielding the following deterministic choice for $\underline{\lambda}_t$:

$$\underline{\lambda}_t^{\text{det}} = (\tau_t\sqrt{\omega_0}, \ -\tau_t\sqrt{\omega_1}, \ \ldots, \ , -\tau_t\sqrt{\omega_t}), \quad t \geq 0. \tag{3.51}$$

At the end of each iteration $t \geq 1$, these coefficients are used to first produce $\mathcal{Z}_t^{\text{comb}}$, which is then used to compute $\mathsf{stat}^t = \tau_t\mathcal{Z}_t^{\text{comb}} + \beta^t$. The updated estimate of the message vector in step $(t+1)$ is $\beta^{t+1} = \eta_t(\mathsf{stat}^t)$, where η_t is defined in (3.23).

The performance of the adaptive successive decoder with deterministic coefficients in (3.51) is given by the following theorem.

Theorem 3.2. *[26, Lemma 11] [27, Lemma 7] Consider a SPARC with a rate $R < \mathcal{C}$, parameters (n, L, M) chosen according to (1.2), and power allocation $P_\ell \propto e^{-2\mathcal{C}\ell/L}$. For $t \geq 1$, let*

$$\mathcal{A}_t := \left\{ \left| \frac{1}{nP}\beta^*\beta^t - x_t \right| > \epsilon \right\} \cup \left\{ \left| \frac{1}{nP}\|\beta^t\|^2 - x_t) \right| > \epsilon \right\},$$

where x_t is defined by the state evolution recursion in (3.25) and (3.25). Then,

$$\mathbb{P}\{\cup_{k=1}^t \mathcal{A}_k\} \leq \sum_{k=1}^t 6(k+1) \exp\left(\frac{-2L\epsilon^2}{c^2(\log M/R)^{2k-1}} \right), \tag{3.52}$$

where $c^2 = \max_\ell LP_\ell/P$, which is a constant close to $2\mathcal{C}(1 + \mathsf{snr})/\mathsf{snr}$ for large L.

We run the decoder for T steps, where T can be determined using the SE recursion in (3.25) as the minimum number of steps after which $1 - x_T$ is below a specified small value δ. Or, using the asymptotic SE characterization in Lemma 3.4, we can take $T = T^* = \left\lceil \frac{2\mathcal{C}}{\log(\mathcal{C}/R)} \right\rceil$. The large deviations bound for $\frac{1}{nP}\beta^*\beta^T$ and $\frac{1}{nP}\|\beta^T\|^2$ in (3.52) can then be translated into a bound on the excess section error rate. We defer the explanation of how this is done to the next section where we analyze the AMP decoder. (See Eq. (3.120) and the surrounding discussion.)

As an alternative to the deterministic coefficients of combination, Cho and Barron [27, 26] propose another method of choosing coefficients based on the Cholesky decomposition of $B^*B = RR^*$ in (3.45). This method uses the known values of $(\beta^k)^*\beta^m$, $1 \leq k \leq m \leq t$, in the matrix B^*B and estimates based on Lemma 3.5 for the diagonal entries of R to recursively solve for the $(b_0^*\beta^t, \ldots, b_{t-1}^*\beta^t)$.

These resulting values are then used to generate $\mathcal{Z}_t^{\text{comb}}$ via (3.40). The reader is referred to [27, Sec. 4.3] or [26, Sec. 4.3] for details of the Cholesky decomposition based estimates and the corresponding performance analysis.

3.4 Approximate Message Passing (AMP) decoder

Approximate message passing (AMP) refers to a class of algorithms [36, 83, 19, 20, 72, 91, 35] that are Gaussian or quadratic approximations of loopy belief propagation algorithms (e.g., min-sum, sum-product) on dense factor graphs. In its basic form [36, 20], AMP gives a fast iterative algorithm to solve the LASSO [109] under certain conditions on the design matrix. The LASSO is the following convex optimization problem. Given a matrix $A \in \mathbb{R}^{n \times N}$, an observation vector $y \in \mathbb{R}^n$, and a scalar $\lambda > 0$, compute

$$\underset{\hat{\beta} \in \mathbb{R}^N}{\arg \min} \; \|y - A\hat{\beta}\|_2^2 + \lambda\|\hat{\beta}\|_1, \tag{3.53}$$

where the ℓ_1-norm is defined as $\|\hat{\beta}\|_1 = \sum_{i=1}^{N} \hat{\beta}_i$. The ℓ_1 penalty added to the least-squares term promotes sparsity in the solution. The LASSO has been widely used in applications such as compressed sensing and sparse linear regression; see, e.g., [110].

When A has i.i.d. entries drawn from a Gaussian or sub-Gaussian distribution, AMP has been found to converge to the LASSO solution (3.53) faster than the best competing solvers (based on first-order convex optimization). This is because AMP takes advantage of the distribution of the matrix A, unlike generic convex optimization methods. The AMP also yields sharp results for the asymptotic risk of LASSO with Gaussian matrices [20].

For SPARCs, recall from (2.2) that the optimal decoder solves the optimization problem

$$\hat{\beta}_{\text{opt}} = \underset{\hat{\beta} \in \mathcal{B}_{M,L}}{\arg \min} \; \|y - A\hat{\beta}\|^2. \tag{3.54}$$

One cannot directly use the LASSO-AMP of [36, 20] for SPARC decoding as it does not use the prior knowledge about β, i.e., the knowledge that β has exactly one non-zero value in each section, with the values of the non-zeros also being known.

We start with the factor graph for the model $y = A\beta + w$, where $\beta \in \mathcal{B}_{M,L}(P_1, \ldots, P_L)$. Each row of A corresponds to a constraint (factor) node, while each column corresponds to a variable node. We use the indices a, b to denote factor nodes, and indices i, j to denote variable nodes. The AMP algorithm is obtained via a first-order approximation to the following message passing updates that iteratively computes estimates of β from y. For $i \in [N]$, $a \in [n]$, set $\beta_{j \to a}^0 = 0$, and compute the following for $t \geq 0$:

$$z_{a \to i}^t = y_a - \sum_{j \in [N] \setminus i} A_{aj} \beta_{j \to a}^t, \tag{3.55}$$

$$\beta_{i \to a}^{t+1} = \eta_i^t \left(\text{stat}_{a \to i} \right), \tag{3.56}$$

39

Step 0: Initialize $\beta^0 = 0$ and $z^{-1} = 0$.

Step $t + 1$, for $0 \le t \le (T - 1)$: Compute

$$z^t = y - A\beta^t + \frac{z^{t-1}}{\tau_{t-1}^2}\left(P - \frac{\|\beta^t\|^2}{n}\right), \qquad (3.59)$$

$$\mathsf{stat}^t = A^* z^t + \beta^t, \qquad (3.60)$$

$$\beta^{t+1} = \eta_t(\mathsf{stat}^t). \qquad (3.61)$$

where the constants $(\tau_t^2)_{t \ge 0}$ required in (3.59) and (3.61) are given by the SE recursion described in (3.25). Instead of pre-computing τ_t^2, it can also be estimated online as $\|z^t\|^2/n$ (see Sec. 4.1.2). The number of iterations T is determined either using the state evolution recursion as discussed on p. 38, or using the termination criterion in Sec. 4.1.2.

Figure 3.5: Approximate Message Passing (AMP) Decoder.

where $\eta_i^t(\cdot)$ is the estimation function defined in (3.23), and for $i \in \sec(\ell)$, the entries of the test statistic $\mathsf{stat}_{i \to a} \in \mathbb{R}^M$ are defined as

$$
\begin{aligned}
(\mathsf{stat}_{a \to i})_i &= \sum_{b \in [n] \backslash a} A_{bi} z_{b \to i}^t, \\
(\mathsf{stat}_{a \to i})_j &= \sum_{b \in [n]} A_{bj} z_{b \to j}^t, \quad j \in \sec(\ell) \backslash i.
\end{aligned}
\qquad (3.57)
$$

As before, the update function (3.56) is based on the assumption that $\mathsf{stat}_{a \to i} \approx \beta + \tau_t Z$. In (3.55), note that the dependence of $z_{a \to i}^t$ on i is only due to the term $A_{ai}\beta_{i \to a}^t$ being excluded from the sum. Similarly, in (3.56) the dependence of $\beta_{i \to a}^t$ on a is due to excluding the term $A_{ai}z_{a \to i}^t$ from the argument. We therefore write

$$z_{a \to i}^t = z_a^t + \delta z_{a \to i}^t, \quad \text{and} \quad \beta_{i \to a}^{t+1} = \beta_i^{t+1} + \delta\beta_{i \to a}^{t+1}. \qquad (3.58)$$

Using a first-order Taylor approximation for the updates (3.55) and (3.56) around the terms z_a^t and β_i^{t+1} and simplifying yields the AMP decoding algorithm which produces iterates (z^t, β^{t+1}) in each iteration. The AMP algorithm is described in Fig. 3.5. (See [95, Appendix A] for details of the derivation.)

The vector z^t in (3.59) is a modified residual: it consists of the standard residual $y - A\beta^t$, plus an extra term $\frac{z^{t-1}}{\tau_{t-1}^2}(P - \frac{\|\beta^t\|^2}{n})$. This extra 'Onsager' term is crucial to ensuring that stat^t has the desired distributional property. To get some intuition about the role of the Onsager term, we express stat^t as

$$
\begin{aligned}
\mathsf{stat}^t = A^* z^t + \beta^t &= A^*(y - A\beta^t) + \beta^t + \frac{A^* z^{t-1}}{\tau_{t-1}^2}\left(P - \frac{\|\beta^t\|^2}{n}\right) \\
&= \beta + A^* w + (\mathsf{I} - A^* A)(\beta^t - \beta) + \frac{A^* z^{t-1}}{\tau_{t-1}^2}\left(P - \frac{\|\beta^t\|^2}{n}\right)
\end{aligned}
\qquad (3.62)
$$

40

We can interpret the second and third terms on the RHS of (3.62) as noise terms added to β. The term A^*w is a random vector independent of β with i.i.d $\mathcal{N}(0, \sigma^2)$ entries. For the next term, the entries of the symmetric matrix $(I - A^*A)$ can be shown to be approximately $\mathcal{N}(0, \frac{1}{n})$, with distinct entries being approximately pairwise independent. Therefore, *if* the $(\beta^t - \beta)$ were independent of A, then the vector $(I - A^*A)(\beta^t - \beta)$ would be approximately i.i.d.$\sim \mathcal{N}(0, \frac{\|\beta^t - \beta\|^2}{n})$; consequently the second and third terms of (3.62) combined would be close to standard normal with variance $\sigma^2 + \frac{\|\beta^t - \beta\|^2}{n} \approx \tau_t^2$. However, $(\beta^t - \beta)$ is not independent of A, since A is used to generate $\beta^1, \ldots, \beta^t$. The role of the last term in (3.62) is to asymptotically cancel the correlation between A and $(\beta^t - \beta)$, so that stat^t is well approximated as $\beta + \tau_t Z$. This intuition is made precise in the analysis of the AMP decoder in the next subsection.

3.4.1 Analysis of the AMP decoder

We now obtain a non-asymptotic bound on error performance of the AMP decoder. To do this, we first need a lower bound on how much the state evolution parameter x_t increases in each iteration of the algorithm.

Lemma 3.6. *[97] Let $\delta \in (0, \min\{\Delta_R, \frac{1}{2}\}]$, where $\Delta_R := (\mathcal{C} - R)/\mathcal{C}$. Let $f(M) := \frac{M^{-\kappa_2 \delta^2}}{\delta \sqrt{\log M}}$, where κ_2 is the universal constant in Lemma 3.3(b). Consider the sequence of state evolution parameters $x_0 = 0, x_1, \ldots$ computed according to (3.25) –(3.26) with the exponentially decaying power allocation in (3.2). For sufficiently large L, M, we have:*

$$x_1 \geq \chi_1 := (1 - f(M)) \frac{P + \sigma^2}{P} \left(1 - \frac{(1 + \delta/2)R}{\mathcal{C}} - \frac{5}{L}\right), \tag{3.63}$$

and for $t > 1$:

$$
\begin{aligned}
x_t &- x_{t-1} \\
&\geq \chi := (1 - f(M)) \left[\frac{\sigma^2}{P}\left(1 - \frac{(1 + \delta/2)R}{\mathcal{C}}\right) - f(M)\frac{(1 + \delta/2)R}{\mathcal{C}}\right] \\
&\quad - \frac{5(1 + \sigma^2/P)}{L},
\end{aligned}
\tag{3.64}
$$

until x_t reaches (or exceeds) $(1 - f(M))$.

Proof. In Section 3.6.2. □

Number of iterations and the gap from capacity We want the lower bounds χ_1 and χ in (3.63) and (3.64) to be strictly positive and depend only on the gap from capacity $\Delta_R = (\mathcal{C} - R)/\mathcal{C}$ as $M, L \to \infty$. For all $\delta \in (0, \Delta_R]$, we have

$$\left(1 - \frac{(1 + \delta/2)R}{\mathcal{C}}\right) \geq \left(1 - \left(1 + \frac{\Delta_R}{2}\right)(1 - \Delta_R)\right) = \frac{\Delta_R + \Delta_R^2}{2}. \tag{3.65}$$

Therefore, the quantities on the RHS of (3.63) and (3.64) can be bounded from below as

$$\chi_1 \geq (1 - f(M)) \frac{P + \sigma^2}{P} \left(\frac{\Delta_R + \Delta_R^2}{2} - \frac{5}{L} \right), \tag{3.66}$$

$$\chi \geq (1 - f(M)) \left[\frac{\sigma^2}{P} \left(\frac{\Delta_R + \Delta_R^2}{2} \right) - f(M) \right] - \frac{5(1 + \sigma^2/P)}{L}. \tag{3.67}$$

We take $\delta = \Delta_R$, which[1] gives the smallest value for $f(M)$. We denote this value by

$$f_R(M) := \frac{M^{-\kappa_2 \Delta_R^2}}{\Delta_R \sqrt{\log M}}. \tag{3.68}$$

From (3.67), if $f_R(M)/\Delta_R \to 0$ as $M \to \infty$, then $\frac{\sigma^2}{P} \left(\frac{\Delta_R + \Delta_R^2}{2} \right)$ will be the dominant term in χ for large enough L, M. The condition $f_R(M)/\Delta_R \to 0$ will be satisfied if we choose Δ_R such that

$$\Delta_R \geq \sqrt{\frac{\log \log M}{\kappa_2 \log M}}, \tag{3.69}$$

where κ_2 is the universal constant of Lemma 3.4. From here on, we assume that Δ_R satisfies (3.69).

Let T be the number of iterations until x_t exceeds $(1 - f_R(M))$. We run the AMP decoder for T iterations, where

$$T := \min_t \{t : x_t \geq 1 - f_R(M)\} \overset{(a)}{\leq} \frac{(1 - f_R(M))}{\chi}$$

$$\overset{(b)}{=} \frac{P/\sigma^2}{(\Delta_R + \Delta_R^2)/2}(1 + o(1)), \tag{3.70}$$

where $o(1) \to 0$ as $M, L \to \infty$. In (3.69), inequality (a) holds for sufficiently large L, M due to Lemma 3.6, which shows for large enough L, M, the x_t value increases by at least χ in each iteration. The equality (b) follows from the lower bound on χ in (3.67), and because $f(M)/\Delta_R = o(1)$.

After running the decoder for T iterations, the decoded message $\hat{\beta}$ is obtained by setting the maximum of β^T in each section $\ell \in [L]$ to $\sqrt{nP_\ell}$ and the remaining entries to 0. From (3.70), we see that the number of iterations T increases as R approaches $\mathcal{C}$. The definition of T guarantees that $x_T \geq (1 - f_R(M))$. Therefore, using $\tau_T^2 = \sigma + P(1 - x_T)$ we have

$$\sigma^2 \leq \tau_T^2 \leq \sigma^2 + Pf_R(M). \tag{3.71}$$

Performance of the AMP decoder   The main result is a bound on the probability of the section error rate exceeding any fixed $\epsilon > 0$.

---

[1] As Lemma 3.4 assumes that $\delta \in (0, \min\{\frac{1}{2}, \Delta_R\}]$, by taking $\delta = \Delta_R$ we have assumed that $\Delta_R \leq \frac{1}{2}$, i.e., $R \geq \mathcal{C}/2$. This assumption can be made without loss of generality — as the probability of error increases with rate, the large deviations bound of Theorem 3.3 evaluated for $\Delta_R = \frac{1}{2}$ applies for all R such that $\Delta_R < \frac{1}{2}$.

Theorem 3.3. *[97] Fix any rate $R < C$. Consider a rate R SPARC S_n with block length n, design matrix parameters L and M determined according to (1.2), and an exponentially decaying power allocation given by (3.2). Furthermore, assume that M is large enough that*

$$\Delta_R \geq \sqrt{\frac{\log \log M}{\kappa_2 \log M}},$$

where κ_2 is the universal constant used in Lemmas 3.3(b) and 3.4. Fix any $\epsilon > \frac{2snr}{C} f_R(M)$, where $f_R(M) := \frac{M^{-\kappa_2 \Delta_R^2}}{\Delta_R \sqrt{\log M}}$.

Then, for sufficiently large L, M, the section error rate of the AMP decoder satisfies

$$P\left(\mathcal{E}_{sec}(S_n) > \epsilon\right) \leq K_T \exp\left\{\frac{-\kappa_T L}{(\log M)^{2T-1}} \left(\frac{\epsilon \sigma^2 C}{2} - P f_R(M)\right)^2\right\}, \tag{3.72}$$

where T is defined in (3.70). The constants κ_T and K_T in (3.72) are given by $\kappa_T = [c^{2T}(T!)^{17}]^{-1}$ and $K_T = C^{2T}(T!)^{11}$ where $c, C > 0$ are universal constants (not depending on AMP parameters L, M, n, or ϵ) but are not explicitly specified.

Remark 3.3. *The dependence of the constants K_T, κ_T on $T!$ is due to the induction-based proof of a key concentration result used in the proof (Lemma 3.10). These constants have not been optimized, but we believe that the dependence of these constants on $T!$ is inevitable in any induction-based proof of the result.*

3.4.2 Error exponent and gap from capacity with AMP decoding

In this subsection we consider the behavior of the bound in Theorem 3.3 in two different regimes. The first is where $R < C$ is held constant as $L, M \to \infty$ (with $n = L \log M / R$) — this is the so-called "error exponent" regime. In this case, Δ_R is of constant order, so $f_R(M)$ in (3.68) decays polynomially with growing M. The other regime is where R approaches C as $L, M \to \infty$ (equivalently, Δ_R shrinks to 0), while ensuring that the error probability remains small or goes to 0. Here, (3.69) specifies that Δ_R should be of order at least $\sqrt{\frac{\log \log M}{\log M}}$.

Error exponent   For any ensemble of codes, the error exponent specifies how the codeword error probability decays with growing code length n for a fixed $R < C$ [46]. In the SPARC setting, we wish to understand how the bound on the probability of excess section error rate in Theorem 3.3 decays with n for fixed values of $\epsilon > 0$ and $R < C$. (As explained in Remark 5 following Theorem 3.3, concatenation using an outer code can be used to extend the result to the codeword error probability.) With optimal encoding, it was shown in [16] that the probability of excess section error rate decays exponentially in $n \min\{\epsilon \Delta, \Delta^2\}$, where $\Delta = (C - R)$. For the AMP decoder, we consider two choices for (M, L) in terms of n to illustrate the trade-offs involved:

1. $M = L^a$, for some constant $a > 0$. Then, (1.2) implies that $L = \Theta(\frac{n}{\log n})$ and $M = \Theta((\frac{n}{\log n})^a)$. Therefore, the bound in Theorem 3.3 decays exponentially in $n/(\log n)^{2T}$.

2. $L = \kappa n / \log \log n$, for some constant κ, which implies $M = \frac{R}{\kappa} \log n$. With this choice the bound in Theorem 3.3 decays exponentially in $n/(\log \log n)^{2T}$.

Note from (3.70) that for a fixed $R < C$, the number of AMP iterations T is an $\Theta(1)$ quantity that does not grow with L, M, or n. The excess section error rate decays more rapidly with n for the second choice, but this comes at the expense of much smaller M (for a given n). Therefore, the first choice allows for a much smaller target section error rate (due to smaller $f_R(M)$), but has a larger probability of deviation from the target. One can also compare the two cases in terms of decoding complexity, which is $O(nMLT)$ with Gaussian design matrices. The complexity in the first case is $O(n^{2+a}/(\log n)^{1+a})$, while in the second case it is $O(n^2 \log n / \log \log n)$.

Gap from capacity   We now consider how fast R can approach the capacity C with growing n, so that the probability of excess section error rate still decays to zero. Recall that lower bound on the gap from capacity is already specified by (3.69): for the state evolution parameter x_T to converge to 1 with growing M (predicting reliable decoding), we need $\Delta_R \geq \sqrt{\frac{\log \log M}{\kappa_2 \log M}}$. When Δ_R takes this minimum value, the minimum target section error rate $f_R(M)$ in Theorem 3.3 is

$$\underline{f_R}(M) = \frac{\sqrt{\kappa_2}}{\log M \sqrt{\log \log M}}. \tag{3.73}$$

We evaluate the large deviations bound of Theorem 3.3 with Δ_R at the minimum value of $\sqrt{\frac{\log \log M}{\kappa_2 \log M}}$, for $\epsilon > \frac{2\mathsf{snr}}{C} \underline{f_R}(M)$, with $\underline{f_R}(M)$ given in (3.73). From (3.70), we have the bound

$$T \leq \frac{2\mathsf{snr}}{\Delta_R} \leq \kappa_4 \sqrt{\frac{\log M}{\log \log M}} \tag{3.74}$$

for large enough L, M. Then, using Stirling's approximation to write $\log(T!) = T \log T - T + O(\log T)$, Theorem 3.3 yields

$$-\log P\left(\mathcal{E}_{sec}(\mathcal{S}_n) > \epsilon\right) \geq \frac{\kappa_5 L \epsilon^2}{c^{2T}(T!)^{17}(\log M)^{2T-1}} - O(T \log T)$$

$$= \frac{\kappa_5 L \epsilon^2}{\exp\{2T \log c + 17(T \log T - T) + (2T - 1)\log \log M + O(\log T)\}} - O(T \log T)$$

$$\geq \frac{L \epsilon^2}{\exp\left\{\kappa_6 \sqrt{(\log M)(\log \log M)}\left(1 + O(\frac{1}{\log \log M})\right)\right\}} - O\left(\sqrt{(\log M)(\log \log M)}\right) \tag{3.75}$$

where the last inequality above follows from (3.74).

We now evaluate the bound in (3.75) for the case $M = L^a$ considered in Sec 3.4.2. We then we

44

have $L = \Theta(\frac{n}{\log n})$ and $M = \Theta((\frac{n}{\log n})^{\mathsf{a}})$. Substituting these in (3.75), we obtain

$$- \log P\left(\mathcal{E}_{sec}(\mathcal{S}_n) > \epsilon\right) \geq \frac{\kappa_7 n \epsilon^2}{(\log n) \exp\{\kappa_8 \sqrt{(\log n)(\log \log n)}\}}$$

$$= \kappa_7 \exp\left\{\log n - \kappa_8 \sqrt{(\log n)(\log \log n)} - \log \log n\right\} \epsilon^2$$

$$= \kappa n^{1-O\left(\sqrt{\frac{\log \log n}{\log n}} + \frac{\log \log n}{\log n}\right)} \epsilon^2. \tag{3.76}$$

Therefore, for the case $M = L^{\mathsf{a}}$, we can achieve a probability of excess section error rate that decays as $\exp\left\{-\kappa n^{1-O\left(\sqrt{\log \log n/\log n}\right)} \epsilon^2\right\}$, with a gap from capacity (Δ_R) that is of order $\sqrt{\frac{\log \log n}{\log n}}$. Furthermore, from (3.73) we see that ϵ must be of order at least $\frac{1}{\log n \sqrt{\log \log n}}$.

We note that this gap from capacity is of a much larger order than that for polar codes over binary input, symmetric memoryless channels [55]. Guruswami and Xia showed in [55] that for such channels, polar codes of block length n with gap from capacity of order $\frac{1}{n^\mu}$ can achieve a block error probability decaying as $2^{-n^{0.49}}$ with a decoding algorithm whose complexity scales as $n \cdot \text{poly}(\log n)$. (Here $0 < \mu < \frac{1}{2}$ is a universal constant.) For AWGN channels, there is no known coding scheme that provably achieves a polynomial gap to capacity with efficient decoding.

Recall the lower bound on the gap to capacity arises from the condition (3.67) which is required to ensure that the (deterministic) state evolution sequence $x_1, x_2, \ldots$ is guaranteed to increase by at least an amount proportional to Δ_R in each iteration. As described in Remark 3.2 the capacity gap for the iterative hard-decision decoder can be improved to $O(\frac{\log \log M}{\log M})$ by modifying the exponential power allocation to flatten the power allocation for a certain number of of sections at the end. We expect such a modification to yield a similar improvement in the capacity gap for the AMP decoder, but we do not detail this analysis as it is involves additional technical details.

3.5   Comparison of the decoders

All three decoders discussed in this section – the adaptive successive hard-decision decoder, the adaptive successive soft-decision decoder, and the AMP decoder — achieve near-exponential decay of error probability in the regime where $R < \mathcal{C}$ remains fixed. However, the finite length performance of the two soft-decision decoders is significantly better than that of the hard-decision decoder. This is because of the need to control the proliferation of false alarms in hard-decision decoding.

In the regime where $R < \mathcal{C}$ is fixed, the number of iterations also remains fixed. Consequently, the complexity of all three decoders is $O(nML)$. The complexity is determined by the matrix-vector products that need to be computed in each step, using the design matrix $A \in \mathbb{R}^{n \times ML}$. Among the two soft-decision decoders, the AMP decoder has lower per iteration complexity (though still of the same order) as it does not require orthonormalization or Cholesky decomposition to compute the test statistic. In the next chapter, we describe how replacing the Gaussian design matrix with a Hadamard-based design matrix can lead to significant savings in both running time and memory.

In the regime where Δ_R shrinks to 0 with growing M, the decoders discussed in this chapter are

no longer efficient as they require M to increase exponentially in $1/\Delta_R$ (cf. (3.69)). An interesting open question is whether SPARCs can achieve a smaller gap from capacity with efficient decoding. The spatially coupled SPARC discussed in Chapter 5 is a promising candidate, but a fully rigorous analysis of AMP-decoded spatially coupled SPARCs remains open.

3.6 Proofs

3.6.1 Proof of Lemma 3.3

From (3.26), $x(\tau)$ can be written as

$$x(\tau) := \sum_{\ell=1}^{L} \frac{P_\ell}{P} \, \mathcal{E}_\ell(\tau), \tag{3.77}$$

where

$$\mathcal{E}_\ell(\tau) = \mathbb{E}\left[\frac{e^{\frac{\sqrt{nP_\ell}}{\tau} U_1^\ell}}{e^{\frac{\sqrt{nP_\ell}}{\tau} U_1^\ell} + e^{-\frac{nP_\ell}{\tau^2}} \sum_{j=2}^{M} e^{\frac{\sqrt{nP_\ell}}{\tau} U_j^\ell}} \right]. \tag{3.78}$$

The result needs to be proved only for $\xi^* > 0$. (For brevity, we supress the dependence of ξ^* on τ.) Since P_ℓ is non-increasing with ℓ, it is enough[2] to prove that for $\xi \in (0, 1]$,

$$\lim \mathcal{E}_{\lfloor \xi L \rfloor}(\tau) = \begin{cases} 1, & \text{if } \xi < \xi^*, \\ 0, & \text{if } \xi > \xi^*. \end{cases} \tag{3.79}$$

Using the relation $nR = L \ln M$, we can write

$$\frac{nP_{\lfloor \xi L \rfloor}}{\tau^2} = \nu_{\lfloor \xi L \rfloor} \ln M, \quad \text{where} \quad \nu_{\lfloor \xi L \rfloor} = \frac{L P_{\lfloor \xi L \rfloor}}{R \tau^2}.$$

From the definition of ξ^* in the lemma statement and the non-increasing power-allocation, we see that $\lim \nu_{\lfloor \xi L \rfloor} > 2$ for $\xi < \xi^*$, and $\lim \nu_{\lfloor \xi L \rfloor} < 2$ for $\xi > \xi^*$.

For brevity, in what follows we drop the superscripts on $U_j^{\lfloor \xi L \rfloor}$, and denote it by U_j for $j \in [M]$. From (3.78), $\mathcal{E}_{\lfloor \xi L \rfloor}(\tau)$ can be written as

$$
\begin{aligned}
\mathcal{E}_{\lfloor \xi L \rfloor}(\tau) &= \mathbb{E}\left[\frac{e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M} \, U_1}}{e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M} \, U_1} + M^{-\nu_{\lfloor \xi L \rfloor}} \sum_{j=2}^{M} e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M} \, U_j}} \right] \\
&= \mathbb{E}\,\mathbb{E}\left[\frac{e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M} \, U_1}}{e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M} \, U_1} + M^{-\nu_{\lfloor \xi L \rfloor}} \sum_{j=2}^{M} e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M} \, U_j}} \Big| U_1 \right]. \tag{3.80}
\end{aligned}
$$

---

[2]We can also prove that $\lim \mathcal{E}_{\lfloor \xi^* L \rfloor} = \frac{1}{2}$, but we do not need this for the exponentially decaying power allocation since it will only affect a vanishing fraction of sections as L increases. Since $\mathcal{E}_\ell \in [0, 1]$, these sections do not affect the value of $\lim x(\tau)$ in (3.78).

The inner expectation in (3.80) is of the form

$$
\mathbb{E}\left[\frac{e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}\, U_1}}{e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}\, U_1} + M^{-\nu_{\lfloor \xi L \rfloor}} \sum_{j=2}^{M} e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}\, U_j}} \,\Big|\, U_1 \right] = \mathbb{E}_X\left[\frac{c}{c + X}\right],
$$

where $c = \exp\left(\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}\, U_1\right)$ is treated as a positive constant, and the expectation is with respect to the random variable

$$
X := M^{-\nu_{\lfloor \xi L \rfloor}} \sum_{j=2}^{M} \exp\left(\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}\, U_j\right). \tag{3.81}
$$

Case 1: $\xi < \xi^*$. Here we have $\lim \nu_{\lfloor \xi L \rfloor} > 2$. Since $\frac{c}{c+X}$ is a convex function of X, applying Jensen's inequality we get $\mathbb{E}_X[\frac{c}{c+X}] \geq \frac{c}{c+\mathbb{E}X}$. The expectation of X is

$$
\mathbb{E}X = M^{-\nu_{\lfloor \xi L \rfloor}} \sum_{j=2}^{M} \mathbb{E}\left[e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}\, U_j}\right]
$$

$$
\overset{(a)}{=} M^{-\nu_{\lfloor \xi L \rfloor}} (M-1) M^{\nu_{\lfloor \xi L \rfloor}/2} \leq M^{1-\nu_{\lfloor \xi L \rfloor}/2},
$$

with (a) is obtained from the moment generating function of a Gaussian random variable. Therefore,

$$
1 \geq \mathbb{E}_X\left[\frac{c}{c+X}\right] \geq \frac{c}{c+\mathbb{E}X} \geq \frac{c}{c+M^{1-\nu_{\lfloor \xi L \rfloor}/2}}
$$
$$
= \frac{1}{1 + c^{-1} M^{1-\nu_{\lfloor \xi L \rfloor}/2}}. \tag{3.82}
$$

Recalling that $c = \exp\left(\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}\, U_1\right)$, (3.82) implies that

$$
\mathbb{E}_X\left[\frac{e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}\, U_1}}{e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}\, U_1} + X} \,\Big|\, U_1 \right] \geq \frac{1}{1 + M^{1-\nu_{\lfloor \xi L \rfloor}/2} e^{-\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}\, U_1}}. \tag{3.83}
$$

When $\{U_1 > -(\ln M)^{1/4}\}$, the RHS of (3.83) is at least

$$
[1 + M^{1-\nu_{\lfloor \xi L \rfloor}/2} \exp\left((\ln M)^{3/4} \sqrt{\nu_{\lfloor \xi L \rfloor}}\right)]^{-1}.
$$

Using this in (3.80), we obtain that

$$
1 \geq \mathcal{E}_{\lfloor \xi L \rfloor}(\tau) \geq \frac{P(U_1 > -(\ln M)^{1/4})}{1 + M^{1-\nu_{\lfloor \xi L \rfloor}/2} e^{(\ln M)^{3/4}\sqrt{\nu_{\lfloor \xi L \rfloor}}}} \overset{M \to \infty}{\longrightarrow} 1, \tag{3.84}
$$

since $\lim \nu_{\lfloor \xi L \rfloor} > 2$. Hence $\mathcal{E}_{\lfloor \xi L \rfloor} \to 1$ when $\lim \nu_{\lfloor \xi L \rfloor} > 2$.

Case 2: $\xi > \xi^*$. Here we have $\lim \nu_{\lfloor \xi L \rfloor} < 2$. The random variable X in (3.81) can be bounded from below as follows.

$$
X \geq M^{-\nu_{\lfloor \xi L \rfloor}} \max_{j \in \{2,\dots,M\}} e^{\sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}\, U_j}
$$
$$
= M^{-\nu_{\lfloor \xi L \rfloor}} e^{[\max_{j \in \{2,\dots,M\}} U_j] \sqrt{\nu_{\lfloor \xi L \rfloor} \ln M}}. \tag{3.85}
$$

47

Using standard bounds for the standard normal distribution, it can be shown that

$$P\left(\max_{j\in\{2,\dots,M\}} U_j < \sqrt{2\ln M}(1-\epsilon)\right) \le e^{-M^{\epsilon(1-\epsilon)}}, \tag{3.86}$$

for $\epsilon = \omega\left(\frac{\ln\ln M}{\ln M}\right)$.[3] Combining (3.86) and (3.85), we obtain that

$$\begin{aligned}
\exp(-M^{\epsilon(1-\epsilon)}) &\ge P\left(\max_{j\in\{2,\dots,M\}} U_j < \sqrt{2\ln M}(1-\epsilon)\right)\\
&\ge P\left(X < M^{-\nu_{\lfloor\xi L\rfloor}} e^{\sqrt{2\ln M}(1-\epsilon)\sqrt{\nu_{\lfloor\xi L\rfloor}\ln M}}\right)\\
&= P\left(X < M^{\sqrt{2\nu_{\lfloor\xi L\rfloor}}(1-\epsilon)-\nu_{\lfloor\xi L\rfloor}}\right).
\end{aligned}$$

Since $\lim \nu_{\lfloor\xi L\rfloor} < 2$ and $\epsilon > 0$ can be an arbitrarily small constant, there exists a strictly positive constant δ such that $\delta < \sqrt{2\nu_{\lfloor\xi L\rfloor}}(1-\epsilon)-\nu_{\lfloor\xi L\rfloor}$ for all sufficiently large L. Therefore, for sufficiently large M, the expectation in (3.6.1) can be bounded as

$$\begin{aligned}
\mathbb{E}_X\left[\frac{c}{c+X}\right] &\le P(X < M^\delta)\cdot 1 + P(X \ge M^\delta)\cdot\frac{c}{c+M^\delta}\\
&\le e^{-M^{\epsilon(1-\epsilon)}} + 1\cdot\frac{c}{c+M^\delta} \le \frac{2}{1+c^{-1}M^\delta}.
\end{aligned} \tag{3.87}$$

Recalling that $c = \exp\left(\sqrt{\nu_{\lfloor\xi L\rfloor}\ln M}\, U_1\right)$, and using the bound of (3.87) in (3.80), we obtain

$$\begin{aligned}
\mathcal{E}_{\lfloor\xi L\rfloor}(\tau) &\le \mathbb{E}\left[\frac{2}{1+M^\delta e^{-\sqrt{\nu_{\lfloor\xi L\rfloor}\ln M}\, U_1}}\right]\\
&\le P(U_1 > (\ln M)^{1/4})\cdot 2 + \frac{2P(U_1 \le (\ln M)^{1/4})}{1+M^\delta e^{-\sqrt{\nu_{\lfloor\xi L\rfloor}}(\ln M)^{3/4}}}\\
&\overset{(a)}{\le} 2e^{-\frac{1}{2}(\ln M)^{1/2}} + 1\cdot\frac{2}{1+e^{\delta\ln M-\sqrt{\nu_{\lfloor\xi L\rfloor}}(\ln M)^{3/4}}}\\
&\overset{(b)}{\longrightarrow} 0 \text{ as } M\to\infty.
\end{aligned} \tag{3.88}$$

In (3.88), (a) is obtained using the bound $\Phi(x) < \exp(-x^2/2)$ for $x \ge 0$, where $\Phi(\cdot)$ is the Gaussian cdf; (b) holds since δ and $\lim \nu_{\lfloor\xi L\rfloor}$ are both positive constants.

This proves that $\mathcal{E}_{\lfloor\xi L\rfloor}(\tau) \to 0$ when $\lim \nu_{\lfloor\xi L\rfloor} < 2$. The proof of the lemma is complete since we have proved both statements in (3.79).

3.6.2 Proof of Lemma 3.6

We will use the following lower bound on the function $x(\tau)$ in (3.26).

---

[3]Recall that $f(n) = \omega(g(n))$ if for each $k > 0$, $|f(n)|/|g(n)| \ge k$ for all sufficiently large n.

Lemma 3.7. *[97, Lemma 2.1] Consider the exponential power allocation power allocation in (3.2), and let $\nu_\ell := LP_\ell/(R\tau^2)$. Then $x(\tau) \geq x_L(\tau)$, where for sufficiently large M and any $\delta \in (0, \frac{1}{2})$,*

$$x_L(\tau) \geq \left(1 - \frac{M^{-\kappa_1\delta^2}}{\delta\sqrt{\log M}}\right)\sum_{\ell=1}^{L}\frac{P_\ell}{P}\mathbf{1}\left\{\nu_\ell > 2 + \delta\right\} \tag{3.89}$$

$$+ \frac{1}{4}\sum_{\ell=1}^{L}\frac{P_\ell}{P}\mathbf{1}\left\{2\left(1 - \frac{\kappa_2}{\sqrt{\log M}}\right) \leq \nu_\ell \leq 2 + \delta\right\}, \tag{3.90}$$

where κ_1, κ_2 are universal positive constants.

Let $x_{t-1} = x < (1 - f(M))$. We only need to consider the case where $\nu_L < (2 + \delta)$, because otherwise all the $\{\nu_\ell\}_{\ell\in[L]}$ values are at least $(2 + \delta)$, and (3.90) guarantees that $x_t \geq (1 - f(M))$.

With $x_{t-1} = x$, we have $\tau_{t-1}^2 = \sigma^2 + P(1 - x)$. Therefore, from (3.32) we have

$$\nu_\ell = \frac{LP_\ell}{R\tau_{t-1}^2} = \frac{\tau_0^2}{R\tau_{t-1}^2}L((1 + \mathsf{snr})^{1/L} - 1)(1 + \mathsf{snr})^{-\ell/L}, \quad \ell \in [L]. \tag{3.91}$$

Using this in (3.90), we have

$$x_t \geq (1 - f(M))\sum_{\ell=1}^{L}\frac{P_\ell}{P}\mathbf{1}\left\{\nu_\ell > 2 + \delta\right\}$$

$$\overset{(a)}{=} (1 - f(M))\sum_{\ell=1}^{L}\frac{P_\ell}{P}\mathbf{1}\left\{\frac{\ell}{L} < \frac{1}{2\mathcal{C}}\log\left(\frac{L((1 + \mathsf{snr})^{1/L} - 1)\tau_0^2}{(2 + \delta)R\tau_{t-1}^2}\right)\right\}$$

$$\overset{(b)}{\geq} (1 - f(M))\sum_{\ell=1}^{L}\frac{P_\ell}{P}\mathbf{1}\left\{\frac{\ell}{L} \leq \frac{1}{2\mathcal{C}}\log\left(\frac{2\mathcal{C}\tau_0^2}{(2 + \delta)R\tau_{t-1}^2}\right)\right\}$$

$$\overset{(c)}{\geq} (1 - f(M))\frac{P + \sigma^2}{P}\left[1 - \exp\left\{-\log\left(\frac{2\mathcal{C}\tau_0^2}{(2 + \delta)R\tau_{t-1}^2}\right) + \frac{2\mathcal{C}}{L}\right\}\right]$$

$$\overset{(d)}{\geq} (1 - f(M))\frac{P + \sigma^2}{P}\left[1 - \frac{(2 + \delta)R\tau_{t-1}^2}{2\mathcal{C}\tau_0^2} - \frac{5}{L}\right]. \tag{3.92}$$

In the above, (a) is obtained using the expression for ν_ℓ in (3.91), while (b) by noting that $L((1 + \mathsf{snr})^{1/L} - 1) = L(e^{2\mathcal{C}/L} - 1) \geq 2\mathcal{C}$. Inequality (c) is obtained by using the geometric series formula: for any $\xi \in (0, 1)$, we have

$$\sum_{\ell=1}^{\lfloor\xi L\rfloor}P_\ell = (P + \sigma^2)(1 - e^{-2\mathcal{C}\lfloor\xi L\rfloor/L}) \geq (P + \sigma^2)(1 - e^{-2\mathcal{C}\xi}e^{2\mathcal{C}/L}).$$

Inequality (d) uses $e^{2\mathcal{C}/L} \leq 1 + 4\mathcal{C}/L$ for large enough L. Substituting $\tau_{t-1}^2 = \sigma^2 + P(1 - x)$, (3.92)

49

implies

$$x_t - x \geq (1 - f(M)) \frac{P + \sigma^2}{P} \left(1 - \frac{5}{L} \right)$$

$$- (1 - f(M)) \frac{(1 + \delta/2)R}{\mathcal{C}} \left(\frac{P + \sigma^2}{P} - x \right) - x$$

$$= (1 - f(M)) \frac{P + \sigma^2}{P} \left(1 - \frac{(1 + \delta/2)R}{\mathcal{C}} - \frac{5}{L} \right)$$

$$- x \left(1 - (1 - f(M)) \frac{(1 + \delta/2)R}{\mathcal{C}} \right). \tag{3.93}$$

Since $\delta < (\mathcal{C} - R)/\mathcal{C}$, the term $\frac{(1+\delta/2)R}{\mathcal{C}}$ is strictly less than 1, and the RHS of (3.93) is strictly decreasing in x. Using the upper bound of $x < (1 - f(M))$ in (3.93) and simplifying, we obtain

$$x_t - x \geq (1 - f(M)) \frac{\sigma^2}{P} \left(1 - \frac{(1 + \delta/2)R}{\mathcal{C}} \right)$$

$$- f(M)(1 - f(M)) \frac{(1 + \delta/2)R}{\mathcal{C}} - \frac{5(1 + \sigma^2/P)}{L}. \tag{3.94}$$

This completes the proof for $t > 1$. For $t = 1$, we start with $x = 0$, and we get the slightly stronger lower bound of χ_1 by substituting $x = 0$ in (3.93).

3.6.3 Proof Sketch of Theorem 3.3

The main ingredients in the proof of Theorem 3.3 are two technical lemmas (Lemma 3.8 and Lemma 3.10). After laying down some definitions and notation that will be used in the proof, we state the two lemmas and use them to prove Theorem 3.3.

Definitions and notation for the proof. For consistency with earlier analyses of AMP, we use notation similar to [19, 95]. Define the following column vectors recursively for $t \geq 0$, starting with $\beta^0 = 0$ and $z^0 = y$.

$$h^{t+1} := \beta_0 - (A^* z^t + \beta^t), \qquad q^t := \beta^t - \beta_0,$$
$$b^t := w - z^t, \qquad m^t := -z^t. \tag{3.95}$$

Recall that β_0 is the message vector chosen by the transmitter. The vector h^{t+1} is the noise in the effective observation $A^* z^t + \beta^t$, while q^t is the error in the estimate β^t. A key ingredient of the proof is showing that h^{t+1} and m^t are approximately i.i.d. $\mathcal{N}(0, \tau_t^2)$, while b^t is approximately i.i.d. $\mathcal{N}(0, \tau_t^2 - \sigma^2)$.

Define $\mathscr{S}_{t_1, t_2}$ to be the sigma-algebra generated by

$$b^0, ..., b^{t_1-1}, m^0, ..., m^{t_1-1}, h^1, ..., h^{t_2}, q^0, ..., q^{t_2}, \text{ and } \beta_0, w.$$

Lemma 3.8 iteratively computes the conditional distributions $b^t|_{\mathscr{S}_{t,t}}$ and $h^{t+1}|_{\mathscr{S}_{t+1,t}}$. Lemma 3.10 then uses this conditional distributions to show the concentration of the mean squared error $\|q^t\|^2/n$.

For $t \geq 1$, let

$$\lambda_t := \frac{-1}{\tau_{t-1}^2} \left(P - \frac{\|\beta^t\|^2}{n} \right). \tag{3.96}$$

We then have

$$b^t + \lambda_t m^{t-1} = Aq^t, \quad \text{and} \quad h^{t+1} + q^t = A^* m^t, \tag{3.97}$$

which follows from (3.59) and (3.95). From (3.97), we have the matrix equations

$$B_t + [0|M_{t-1}]\Lambda_t = AQ_t \quad \text{and} \quad H_t + Q_t = A^* M_t, \tag{3.98}$$

where for $t \geq 1$,

$$M_t := [m^0 \mid \ldots \mid m^{t-1}], \qquad Q_t := [q^0 \mid \ldots \mid q^{t-1}]$$
$$B_t := [b^0 \mid \ldots \mid b^{t-1}], \quad H_t = [h^1 \mid \ldots \mid h^t], \quad \Lambda_t := \mathrm{diag}(\lambda_0, \ldots, \lambda_{t-1}). \tag{3.99}$$

The notation $[c_1 \mid c_2 \mid \ldots \mid c_k]$ is used to denote a matrix with columns $c_1, \ldots, c_k$. We define M_0, Q_0, B_0, H_0, and Λ_0 to be all-zero vectors.

We use $m_\parallel^t$ and $q_\parallel^t$ to denote the projection of m^t and q^t onto the column space of M_t and Q_t, respectively. Let $\alpha_t := (\alpha_0^t, \ldots, \alpha_{t-1}^t)^*$ and $\gamma_t := (\gamma_0^t, \ldots, \gamma_{t-1}^t)^*$ be the coefficient vectors of these projections, i.e.,

$$m_\parallel^t = \sum_{i=0}^{t-1} \alpha_i^t m^i, \quad q_\parallel^t = \sum_{i=0}^{t-1} \gamma_i^t q^i. \tag{3.100}$$

The projections of m^t and q^t onto the orthogonal complements of M^t and Q^t, respectively, are denoted by

$$m_\perp^t := m^t - m_\parallel^t, \quad q_\perp^t := q^t - q_\parallel^t \tag{3.101}$$

The proof of Lemma 3.10 shows that for large n, the entries of α_t and γ_t concentrate around constants. We now specify these constants. With τ_t^2 and x_t as defined in (3.25) and (3.26), for $t \geq 0$ define

$$\sigma_t^2 := \tau_t^2 - \sigma^2 = P(1 - x_t). \tag{3.102}$$

The concentrating values for γ^t and α^t are

$$\hat{\gamma}^t := (0, \ldots, 0, \sigma_t^2/\sigma_{t-1}^2)^* \in \mathbb{R}^t,$$
$$\hat{\alpha}^t := (0, \ldots, 0, \tau_t^2/\tau_{t-1}^2)^* \in \mathbb{R}^t. \tag{3.103}$$

Let $(\sigma_0^\perp)^2 := \sigma_0^2$ and $(\tau_0^\perp)^2 := \tau_0^2$, and for $t > 0$ define

$$(\sigma_t^\perp)^2 := \sigma_t^2 \left(1 - \frac{\sigma_t^2}{\sigma_{t-1}^2} \right), \quad \text{and} \quad (\tau_t^\perp)^2 := \tau_t^2 \left(1 - \frac{\tau_t^2}{\tau_{t-1}^2} \right). \tag{3.104}$$

Lemma 3.8 (Conditional distribution lemma [95, Lemma 4])**. *For the vectors h^{t+1} and b^t defined in (3.95), the following hold for $1 \leq t \leq T$, provided $n > T$, and M_t and Q_t have full column rank. (We recall that the number of iterations T is defined in (3.70).)*

$$h^1|_{\mathscr{S}_{1,0}} \overset{d}{=} \tau_0 Z_0 + \Delta_{1,0}, \quad \text{and} \quad h^{t+1}|_{\mathscr{S}_{t+1,t}} \overset{d}{=} \frac{\tau_t^2}{\tau_{t-1}^2} h^t + \tau_t^\perp Z_t + \Delta_{t+1,t}, \tag{3.105}$$

$$b^0|_{\mathscr{S}_{0,0}} \overset{d}{=} \sigma_0 Z_0', \quad \text{and} \quad b^t|_{\mathscr{S}_{t,t}} \overset{d}{=} \frac{\sigma_t^2}{\sigma_{t-1}^2} b^{t-1} + \sigma_t^\perp Z_t' + \Delta_{t,t}. \tag{3.106}$$

where $Z_0, Z_t \in \mathbb{R}^N$ and $Z'_0, Z'_t \in \mathbb{R}^n$ are i.i.d. standard Gaussian random vectors that are independent of the corresponding conditioning sigma algebras. The deviation terms are $\Delta_{0,0} = 0$,

$$
\Delta_{1,0} = \left[\left(\frac{\|m^0\|}{\sqrt{n}} - \tau_0 \right) \mathsf{I} - \frac{\|m^0\|}{\sqrt{n}} \mathsf{P}_{q^0} \right] Z_0
$$
$$
+ q^0 \left(\frac{\|q^0\|^2}{n} \right)^{-1} \left(\frac{(b^0)^* m_0}{n} - \frac{\|q^0\|^2}{n} \right),
\tag{3.107}
$$

and for $t > 0$,

$$
\Delta_{t,t} = \sum_{r=0}^{t-2} \gamma_r^t b^r + \left(\gamma_{t-1}^t - \frac{\sigma_t^2}{\sigma_{t-1}^2} \right) b^{t-1} + \left[\left(\frac{\|q_\perp^t\|}{\sqrt{n}} - \sigma_t^\perp \right) \mathsf{I} - \frac{\|q_\perp^t\|}{\sqrt{n}} \mathsf{P}_{M_t} \right] Z'_t
$$
$$
+ M_t \left(\frac{M_t^* M_t}{n} \right)^{-1} \left(\frac{H_t q_\perp^t}{n} - \frac{M_t^{\ *}}{n} \left[\lambda_t m^{t-1} - \sum_{r=1}^{t-1} \lambda_r \gamma_r^t m^{r-1} \right] \right),
\tag{3.108}
$$

$$
\Delta_{t+1,t} = \sum_{r=0}^{t-2} \alpha_r^t h^{r+1} + \left(\alpha_{t-1}^t - \frac{\tau_t^2}{\tau_{t-1}^2} \right) h^t
$$
$$
+ \left[\left(\frac{\|m_\perp^t\|}{\sqrt{n}} - \tau_t^\perp \right) \mathsf{I} - \frac{\|m_\perp^t\|}{\sqrt{n}} \mathsf{P}_{Q_{t+1}} \right] Z_t
$$
$$
+ Q_{t+1} \left(\frac{Q_{t+1}^* Q_{t+1}}{n} \right)^{-1} \left(\frac{B_{t+1}^* m_\perp^t}{n} - \frac{Q_{t+1}^*}{n} \left[q^t - \sum_{i=0}^{t-1} \alpha_i^t q^i \right] \right).
\tag{3.109}
$$

The next lemma uses the representation in Lemma 3.8 to show that for each $t \geq 0$, h^{t+1} is the sum of an i.i.d. $\mathcal{N}(0, \tau_t^2)$ random vector plus a deviation term. Similarly b^t is the sum of an i.i.d. $\mathcal{N}(0, \sigma_t^2)$ random vector and a deviation term.

Lemma 3.9. *For $t \geq 0$, the conditional distributions in Lemma 3.8 can be expressed as*

$$
h^{t+1} |_{\mathscr{S}_{t+1,t}} \overset{d}{=} \tilde{h}^{t+1} + \tilde{\Delta}_{t+1}, \qquad b^t |_{\mathscr{S}_{t,t}} \overset{d}{=} \breve{b}^t + \breve{\Delta}_t,
\tag{3.110}
$$

where

$$
\tilde{h}^{t+1} := \tau_t^2 \sum_{i=0}^t \left(\frac{\tau_i^\perp}{\tau_i^2} \right) Z_i, \qquad \tilde{\Delta}_{t+1} := \tau_t^2 \sum_{i=0}^t \left(\frac{1}{\tau_i^2} \right) \Delta_{i+1,i},
\tag{3.111}
$$

$$
\breve{b}^t := \sigma_t^2 \sum_{i=0}^t \left(\frac{\sigma_i^\perp}{\sigma_i^2} \right) Z'_i, \qquad \breve{\Delta}_t := \sigma_t^2 \sum_{i=0}^t \left(\frac{1}{\sigma_i^2} \right) \Delta_{i,i}.
\tag{3.112}
$$

Here $Z_i \in \mathbb{R}^N$, $Z'_i \in \mathbb{R}^n$ are the independent standard Gaussian vectors defined in Lemma 3.8.

Consequently, $\tilde{h}^{t+1} \overset{d}{=} \tau_t \tilde{Z}_t$, and $\breve{b}^t \overset{d}{=} \sigma_t \breve{Z}_t$, where $\tilde{Z}_t \in \mathbb{R}^N$ and $\breve{Z}_t \in \mathbb{R}^n$ are standard Gaussian random vectors such that for any $j \in [N]$ and $i \in [n]$, the vectors $(\tilde{Z}_{0,j}, \ldots, \tilde{Z}_{t,j})$ and $(\breve{Z}_{0,i}, \ldots, \breve{Z}_{t,i})$ are each jointly Gaussian with

$$
\mathbb{E}[\tilde{Z}_{r,j} \tilde{Z}_{s,j}] = \frac{\tau_s}{\tau_r}, \qquad \mathbb{E}[\breve{Z}_{r,i} \breve{Z}_{s,i}] = \frac{\sigma_s}{\sigma_r} \qquad \text{for } 0 \leq r \leq s \leq t.
\tag{3.113}
$$

Proof. We give the proof for the distributional representation of h^{t+1}, with the proof for b^t being similar. The representation in (3.110) can be directly obtained by using Lemma 3.8 Eq. (3.105) to recursively write h^t in terms of $(h^{t-1}, Z_{t-1}, \Delta_{t,t-1})$, then h^{t-1} in terms of $(h^{t-2}, Z_{t-2}, \Delta_{t-1,t-2})$, and so on.

Using (3.111), we write $\tilde{h}^{t+1} = \tau_t \tilde{Z}_t$, where $\tilde{Z}_t = \tau_t \sum_{i=0}^{t} \left(\frac{\tau_i^\perp}{\tau_i^2} \right) Z_i$ is n Gaussian random vector with i.i.d. entries, with zero mean and variance equal to

$$\tau_t^2 \sum_{i=0}^{t} \frac{(\tau_i^\perp)^2}{\tau_i^4} = \frac{\tau_t^2}{\tau_0^2} + \sum_{i=1}^{t} \left(\frac{\tau_t^2}{\tau_i^2} \right) \left(1 - \frac{\tau_i^2}{\tau_{i-1}^2} \right) = \frac{\tau_t^2}{\tau_0^2} + \sum_{i=1}^{t} \left(\frac{\tau_t^2}{\tau_i^2} - \frac{\tau_t^2}{\tau_{i-1}^2} \right)$$
$$= 1. \tag{3.114}$$

For $j \in [N]$ the covariance between the jth entries of $\tilde{Z}_r$ and $\tilde{Z}_s$, for $0 \le r \le s \le t$, is

$$\mathbb{E}[\tilde{Z}_{r,j} \tilde{Z}_{s,j}] = \tau_r \tau_s \sum_{u=0}^{r} \sum_{v=0}^{s} \left(\frac{\tau_u^\perp}{\tau_u^2} \right) \left(\frac{\tau_v^\perp}{\tau_v^2} \right) \mathbb{E} \left\{ Z_{u_j} Z_{v_j} \right\} \overset{(a)}{=} \tau_r \tau_s \sum_{u=0}^{r} \frac{(\tau_u^\perp)^2}{\tau_u^4}$$
$$\overset{(b)}{=} \frac{\tau_s}{\tau_r}, \tag{3.115}$$

where (a) follows from the independence of Z_{u_j} and Z_{v_j} and (b) from the calculation in (3.114). $\square$

The next lemma shows that the deviation terms in Lemma 3.8 are small, in the sense that their section-wise maximum absolute value and norm concentrate around 0. It also shows that the mean-squared error $\|q^t\|/n = \|\beta - \beta^t\|^2/n$ concentrates around σ_t^2 for $0 \le t \le T$.

Lemma 3.10. *[97] With C, K, c, κ denoting generic positive universal constants, the following large deviations inequalities hold for $0 \le t < T$:*

$$P \left(\left[\frac{1}{L} \sum_{\ell=1}^{L} \max_{j \in sec_\ell} |[\Delta_{t+1,t}]_j| \right]^2 \ge \epsilon \right) \le P \left(\frac{1}{L} \sum_{\ell=1}^{L} \max_{j \in sec_\ell} ([\Delta_{t+1,t}]_j)^2 \ge \epsilon \right)$$
$$\le K C^{2t} (t!)^{11} \exp \left\{ -\frac{\kappa L \epsilon}{(c \log M)^{2t} (t!)^{17}} \right\}, \tag{3.116}$$

$$\tag{3.117}$$

$$P \left(\frac{1}{n} \|\Delta_{t,t}\|^2 \ge \epsilon \right) \le K C^{2t} (t!)^{11} \exp \left\{ -\frac{\kappa L \epsilon^2}{(c \log M)^{2t-1} (t!)^{17}} \right\}, \tag{3.118}$$

$$P \left(\left| \frac{\|q^{t+1}\|^2}{n} - \sigma_{t+1}^2 \right| \ge \epsilon \right) \le K C^{2t} (t!)^{11} \exp \left\{ -\frac{\kappa L \epsilon^2}{(c \log M)^{2t+1} (t!)^{17}} \right\}. \tag{3.119}$$

The proof of Lemma 3.10 can be found in [97, Sec. 5]. The proof is inductive. To prove Theorem 3.3, we only need the concentration result for the squared error $\|q^t\|^2/n$ in (3.119). But the proof

of this result requires concentration results for various inner products and functions involving $\{h^{t+1}, q^t, b^t, m^t\}$, which are proved inductively.

The dependence on t of the probability bounds in Lemma 3.10 is determined by the induction used in the proof: the concentration results for step t depend on those corresponding to all the previous steps. The $t!$ terms in the constants arise due to quantities that can be expressed as a sum of t terms with step indices $1, \ldots, t$, e.g., $\Delta_{t,t}$ and $\Delta_{t+1,t}$ in (3.108) and (3.109). The concentration results for such quantities have $1/t$ and t multiplying the exponent and pre-factor, respectively, in each step t, which results in the $t!$ terms in the bound. Similarly, the C^{2t} and c^{2t} terms arise due to quantities that are the *product* of two terms, for each of which we have a concentration result available from the induction hypothesis.

Proof of Theorem 3.3. The event that the section error rate exceeds ϵ is $\{\mathcal{E}_{sec}(\mathcal{S}_n) > \epsilon\} = \left\{\sum_\ell \mathbf{1}\{\hat{\beta}_\ell \neq \beta_{0_\ell}\} > L\epsilon\right\}$. Recall that the largest entry within each section of β^T is chosen to produce $\hat{\beta}$. Therefore, when a section ℓ is decoded in error, the correct non-zero entry has no more than half the total mass of section ℓ at the termination step T. That is, $\beta^T_{\mathsf{sent}(\ell)} \leq \frac{1}{2}\sqrt{nP_\ell}$ where $\mathsf{sent}(\ell)$ is the index of the non-zero entry in section ℓ of the true message β_0. Since $\beta_{0_{\mathsf{sent}(\ell)}} = \sqrt{nP_\ell}$, we have

$$\mathbf{1}\{\hat{\beta}_\ell \neq \beta_{0_\ell}\} \quad \Rightarrow \quad \|\beta^T_\ell - \beta_{0_\ell}\|^2 \geq \frac{nP_\ell}{4}, \quad \ell \in [L]. \tag{3.120}$$

Hence when $\{\mathcal{E}_{sec}(\mathcal{S}_n) > \epsilon\}$, we have

$$\|\beta^T - \beta_0\|^2 = \sum_{\ell=1}^L \|\beta^T_\ell - \beta_{0_\ell}\|^2 \overset{(a)}{\geq} \sum_{\ell=1}^L \mathbf{1}\{\hat{\beta}_\ell \neq \beta_{0_\ell}\}\frac{nP_\ell}{4}$$

$$\overset{(b)}{\geq} L\epsilon\frac{nP_L}{4} \overset{(c)}{\geq} \frac{n\,\epsilon\,\sigma^2\ln(1+\mathsf{snr})}{4} = \frac{n\epsilon\sigma^2\mathcal{C}}{2}, \tag{3.121}$$

where (a) follows from (3.120); (b) is obtained using the fact that $P_\ell > P_L$ for $\ell \in [L-1]$ for the exponentially decaying power allocation in (3.2); (c) is obtained using the first-order Taylor series lower bound $LP_L \geq \sigma^2\ln(1+\frac{P}{\sigma^2})$. We therefore conclude that

$$\{\mathcal{E}_{sec}(\mathcal{S}_n) > \epsilon\} \quad \Rightarrow \quad \left\{\frac{\|\beta^T - \beta_0\|^2}{n} \geq \frac{\epsilon\sigma^2\mathcal{C}}{2}\right\}, \tag{3.122}$$

where β^T is the AMP estimate at the termination step T.

Now, from (3.119) of Lemma 3.10, we know that for any $\tilde{\epsilon} \in (0,1)$:

$$P\left(\frac{\|\beta^T - \beta_0\|^2}{n} \geq \sigma_T^2 + \tilde{\epsilon}\right) = P\left(\frac{\|q^T\|^2}{n} \geq \sigma_T^2 + \tilde{\epsilon}\right)$$

$$\leq K_T \exp\left\{-\frac{\kappa_T L\tilde{\epsilon}^2}{(\log M)^{2T-1}}\right\}. \tag{3.123}$$

From the definition of T and (3.71), we have $\sigma_T^2 = \tau_T^2 - \sigma^2 \leq P f_R(M)$. Hence, (3.123) implies

$$P\left(\frac{\|\beta^T - \beta_0\|^2}{n} \geq P f_R(M) + \tilde{\epsilon}\right) \leq P\left(\frac{\|\beta^T - \beta_0\|^2}{n} \geq \sigma_T^2 + \tilde{\epsilon}\right)$$

$$\leq K_T \exp\left\{-\frac{\kappa_T L \tilde{\epsilon}^2}{(\log M)^{2T-1}}\right\}. \quad (3.124)$$

Now take $\tilde{\epsilon} = \frac{\epsilon \sigma^2 \mathcal{C}}{2} - P f_R(M)$, noting that this $\tilde{\epsilon}$ is strictly positive whenever $\epsilon > 2\mathsf{snr} f_R(M)/\mathcal{C}$, the condition specified in the theorem statement. Finally, combining (3.122) and (3.124) we obtain

$$P\left(\mathcal{E}_{sec}(\mathcal{S}_n) > \epsilon\right) \leq K_T \exp\left\{-\frac{\kappa_T L}{(\log M)^{2T-1}}\left(\frac{\epsilon \sigma^2 \mathcal{C}}{2} - P f_R(M)\right)^2\right\}.$$

$\square$

Chapter 4

Finite Length Decoding Performance

In this chapter, we investigate the empirical error performance of SPARCs with AMP decoding at finite block lengths. In Section 4.1, we describe how decoding complexity can be reduced by using Hadamard-based design matrices, and how a key parameter of the AMP decoder can be estimated online. In Section 4.2, we show that the choice of power allocation can have a significant impact on decoding performance, and describe a simple algorithm to design a good allocation for a given rate and snr. Section 4.3 discusses how the choice of the code parameters L, M influences finite length error performance. Finally, in Section 4.5 we show how partial outer codes can be used in conjunction with AMP decoding to obtain a steep waterfall in the error rate curves. We compare the error rates of AMP-decoded sparse superposition codes with coded modulation using LDPC codes from the WiMAX standard.

4.1 Reducing AMP decoding complexity

4.1.1 Hadamard-based design matrices

In the sparse regression codes described and analyzed thus far, the design matrix A is chosen to have zero-mean i.i.d. entries, either Gaussian $\sim \mathcal{N}(0, \frac{1}{n})$ or Bernoulli entries drawn uniformly from $\pm\frac{1}{\sqrt{n}}$ as in Sec. 2.3. As discussed in Sec. 3.5, with such matrices the computational complexity of the AMP decoder in (3.59)–(3.61) is $O(LMn)$ when the matrix-vector multiplications $A\beta$ and A^*z^t are performed in the usual way. Additionally, storing A requires $O(LMn)$ memory, which is prohibitive for reasonable code lengths. For example, $L = 1024$, $M = 512$, $n = 9216$ ($R = 1$ bit) requires 18 gigabytes of memory using a double-precision (4-byte) floating point representation, all of which must be accessed twice per iteration.

To reduce decoding complexity, we replace the i.i.d. design matrix with a structured Hadamard-based design matrix, which we denote in this section by A_{H}. With A_{H}, the key matrix-vector multiplications can be performed via a fast Walsh-Hadamard Transform (FWHT)[101]. Moreover, A_{H} can be implicitly defined which greatly reduces the memory required.

We denote the Hadamard matrix of size $2^k \times 2^k$ by H_k. We recall that H_k is a square matrix with ± 1 entries and mutually orthogonal rows, recursively defined as follows. Starting with $H_0 = 1$, for $k \geq 1$,

$$H_k = \begin{pmatrix} H_{k-1} & H_{k-1} \\ H_{k-1} & -H_{k-1} \end{pmatrix}.$$

To construct the design matrix $A_\mathsf{H} \in \mathbb{R}^{n \times ML}$, one option is to take $k = \lceil \log_2(LM) \rceil$ and select n rows uniformly at random from the Hadamard matrix H_k. In this case, the matrix-vector multiplications are performed by embedding the vectors into $\mathbb{R}^{ML}$, and then multiplying by H_k using a FWHT. A more efficient way is to construct each $n \times M$ section of A_H independently from a smaller Hadamard matrix. This is done as follows.

Take $k = \lceil \log_2(\max(n+1, M+1)) \rceil$. Each section of A_H is constructed independently by choosing a permutation of n distinct rows from H_k uniformly at random.[1] The multiplications $A_\mathsf{H} \beta^t$ and $A_\mathsf{H}^* z^t$ are performed by computing $A_{\mathsf{H}\ell} \beta_\ell^t$ and $A_{\mathsf{H}\ell}^* z^t$, for $\ell \in [L]$, where the $n \times M$ matrix $A_{\mathsf{H}\ell}$ is the ℓth section of A_H, and $\beta_\ell^t \in \mathbb{R}^M$ is the ℓth section of β^t. To compute $A_{\mathsf{H}\ell} \beta_\ell^t$, zero-prepend β_ℓ to length 2^k, perform the FWHT, then choose n entries corresponding to the rows in $A_{\mathsf{H}\ell}$. Sum the n-length result from each section to obtain $A_\mathsf{H} \beta^t$. Note that we prepend with 0 because the first column of H_k must always be ignored as it is always all-ones. For $A_{\mathsf{H}\ell}^* z^t$, embed entries from z^t into a 2^k long vector again corresponding to the rows in A_H, with all other entries set to zero, perform the FWHT, and return the last M entries. Concatenate the result from each section to form $A_\mathsf{H}^* z^t$.

The empirical error performance of the AMP decoder with A_H constructed as above is indistinguishable from that of a full i.i.d. matrix. The computational complexity of the decoder is reduced to $O(Ln \log n)$ (in the common case where $n > M$, otherwise it is $O(LM \log M)$). The memory requirements are reduced to $O(LM)$, typically a few megabytes. In comparison, for i.i.d. design matrices, the complexity and memory requirements scale as $O(LMn)$. For reasonable code lengths, this represents around a thousandfold improvement in both time and memory. Furthermore, the easily parallelized structure would enable a hardware implementation to trade off between a slower and smaller series implementation and a faster though larger parallel implementation, potentially leading to significant practical speedups.

4.1.2 Online computation of τ_t^2 and early termination

Recall that these coefficients $(\tau_t^2)_{t \geq 1}$ are required for the AMP update steps (3.59) and (3.61). In the standard implementation, these are recursively computed in advance via the SE equations (3.25) and (3.26). The total number of iterations T is also determined in advance by computing the number of iterations required the SE to converge to its fixed point (to within a specified tolerance). This advance computation is slow as each of the L expectations in (3.26) needs to be computed numerically via Monte-Carlo simulation, for each t.

---

[1]To obtain the desired statistical properties for A, we do not pick the first row of H_k as it is all-ones. The $n+1$ in the definition of k ensures that we still have enough rows left to pick n at random after removing the first, all-one, row; the $M+1$ ensures that we can always have one leading 0 when embedding β so that the first, all-one, column is also never picked.

A simple way to estimate τ_t^2 online during the decoding process is as follows. In each step t, after producing z^t as in (3.55), we estimate

$$\widehat{\tau}_t^2 = \frac{\|z^t\|^2}{n} = \frac{1}{n}\sum_{i=1}^{n} z_i^2. \tag{4.1}$$

The justification for this estimate comes from the analysis of the AMP decoder in [97], which provides a concentration inequality that shows that for large n, $\widehat{\tau}_t^2$ is close to τ_t^2 with high probability. We note that such a similar online estimate has been used previously in various AMP and GAMP algorithms [11, 13, 14, 91].

In addition to being fast, the online estimator permits an interpretation as a measure of SPARC decoding progress and provides a flexible termination criterion for the decoder. Recall from the previous chapter (cf. Section 3.2) that in each step we have

$$\mathsf{stat}^t = \beta^t + A^* z^t \approx \beta + \tau_t Z,$$

where Z is a standard normal random vector independent of β. The online estimator $\widehat{\tau}_t^2$ is found to track $\mathrm{Var}(\mathsf{stat}^t - \beta) = \|\mathsf{stat}^t - \beta\|^2/n$ very accurately, even when this variance deviates significantly from τ_t^2. This indicates that we can use the final value $\widehat{\tau}_T^2$ to accurately estimate the power of the undecoded sections — and thus the number of sections decoded correctly — at runtime. Indeed, $(\widehat{\tau}_T^2 - \sigma^2)$ is an accurate estimate of the total power in the incorrectly decoded sections. This, combined with the fact that the power allocation is non-increasing, allows the decoder to estimate the number of incorrectly decoded sections.

Furthermore, we can use the change in $\widehat{\tau}_t^2$ between iterations to terminate the decoder early. If the value $\widehat{\tau}_t^2$ has not changed between successive iterations, or the change is within some small threshold, then the decoder has stalled and no further iterations are worthwhile. Empirically we find that a stopping criterion with a small threshold (e.g., stop when $\left|\widehat{\tau}_t^2 - \widehat{\tau}_{t-1}^2\right| < P_L$) leads to no additional errors compared to running the decoder for the full iteration count, while giving a significant speedup in most trials. Allowing a larger threshold for the stopping criterion gives even better running time improvements.

All the simulation results reported in this chapter are obtained using Hadamard-based design matrices, the online estimate $\widehat{\tau_t^2}$, and a corresponding early termination criterion.

4.2 Power allocation

Theorem 3.3 shows that for any fixed $R < \mathcal{C}$ and an exponentially decaying power allocation $P_\ell \propto e^{-2\mathcal{C}\ell/L}$, $\ell \in [L]$, the probability of section error of the AMP decoder can be made arbitrarily small for sufficiently large values of the code parameters (n, M, L). However, the error rate of the exponentially decaying allocation is rather high at practical block lengths. This is illustrated in Fig. 4.1. The black curve at the top shows the average section error rate with the exponentially decaying allocation for various rates R with $\mathcal{C} = 2$ bits. The blue curve in the middle shows the average section error rate with two different power allocation schemes, with the code parameters (n, M, L) at each rate.
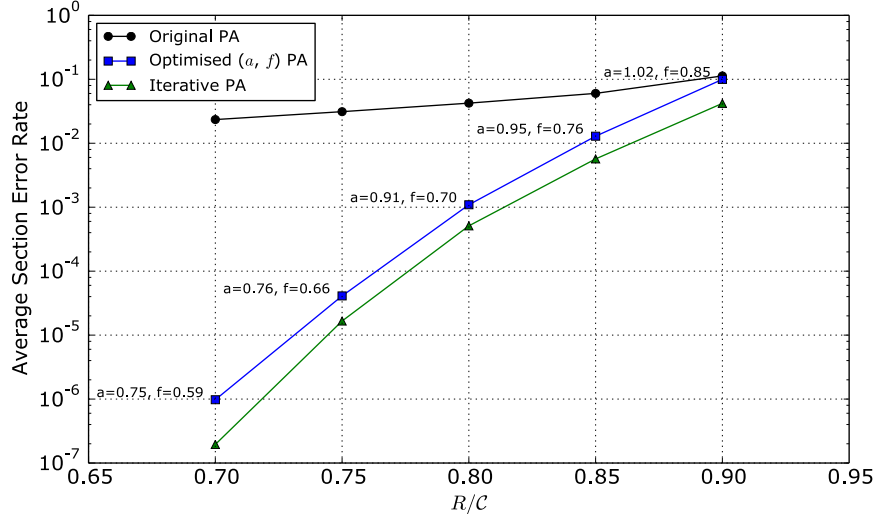
Figure 4.1: Section error rate vs $R/\mathcal{C}$ at $\mathsf{snr} = 15, \mathcal{C} = 2$ bits. The SPARC parameters for all the curves are $M = 512, L = 1024$. The top curve shows the average section error rate of the AMP over 1000 trials with $P_\ell \propto 2^{-2\mathcal{C}\ell/L}$ (with $\mathcal{C}$ in bits). The curve in the middle shows the section error rate using the power allocation in (4.2) with the (a, f) values shown. The bottom curve shows the section error rate with the iterative power allocation scheme described in Section 4.2.1.

It is evident that as we back off from capacity, the power allocation can crucially determine error performance. The reason for the relative poor performance of the exponential allocation at lower rates such as $R = 0.6\mathcal{C}$ and $0.7\mathcal{C}$ is that that it allocates too much power to the initial sections, leaving too little for the final sections to decode reliably. This motivates the following *modified* exponential allocation characterized by two parameters a, f. For $f \in [0, 1]$, let

$$P_\ell = \begin{cases} \kappa \cdot 2^{-2a\mathcal{C}\ell/L}, & 1 \leq \ell \leq fL \\ \kappa \cdot 2^{-2a\mathcal{C}f}, & fL + 1 \leq \ell \leq L \end{cases} \tag{4.2}$$

where the normalizing constant κ ensures that the total power across sections is P. For intuition, first assume that $f = 1$. Then (4.2) implies that $P_\ell \propto 2^{-a2\mathcal{C}\ell/L}$ for $\ell \in [L]$. Setting $a = 1$ recovers the original power allocation of (3.2), while $a = 0$ allocates $\frac{P}{L}$ to each section. Increasing a increases the power allocated to the initial sections which makes them more likely to decode correctly, which in turn helps by decreasing the effective noise variance $\bar{\tau}_t^2$ in subsequent AMP iterations. However, if a is too large, the final sections may have too little power to decode correctly.

Hence we want the parameter a to be large enough to ensure that the AMP gets started on the right track, but not much larger. This intuition can be made precise in the large system limit using Lemma 3.3: recall that for a section ℓ to be correctly decoded in step $(t+1)$, the limit of LP_ℓ must exceed a threshold proportional to $R\bar{\tau}_t^2$. For rates close to $\mathcal{C}$, we need a to be close to 1 for the initial sections to cross this threshold and get decoding started correctly. On the other hand, for rates such as $R = 0.6\mathcal{C}$, $a = 1$ allocates more power than necessary to the initial sections, leading to poor error performance in the final sections.

In addition, we found that the section error rate can be further improved by *flattening* the power allocation in the final sections. For a given a, (4.2) has an exponential power allocation until section

60

fL, and constant power for the remaining $(1 - f)L$ sections. The allocation in (4.2) is continuous, i.e. each section in the flat part is allocated the same power as the final section in the exponential part. Flattening boosts the power given to the final sections compared to an exponentially decaying allocation. The two parameters (a, f) let us trade-off between the conflicting objectives of assigning enough power to the initial sections and ensuring that the final sections have enough power to be decoded correctly.

The middle curve (blue) in Figure 4.1 shows the error performance with this modified allocation. While this allocation improves the section error rate by a few orders of magnitude, it requires costly numerical optimization of a and f. A good starting point is to use $a = f = R/\mathcal{C}$, but further optimization is generally necessary. This motivates the need for a fast power allocation algorithm with fewer tuning parameters.

4.2.1   Iterative power allocation

We now describe a simple iterative algorithm to design a power allocation. The starting point for our power allocation design is the asymptotic expression for the state evolution parameter $x(\tau)$ in Lemma 3.3 (see also the non-asymptotic lower bound in Lemma (3.7)). Here, assuming (L, M, n) are sufficiently large, we use the following approximation:

$$x(\tau) \approx \sum_{\ell=1}^{L} \frac{P_\ell}{P} \mathbf{1} \left\{ LP_\ell > 2R\tau^2 \right\}. \tag{4.3}$$

We note that R in (4.3) is measured in nats. If the effective noise variance after step t is τ_t^2, then (4.3) says that any section ℓ whose normalized power LP_ℓ is larger than the threshold $2R\tau_t^2$ is likely to be decodable in step $(t + 1)$, i.e., in β^{t+1}, the probability mass within the section will be concentrated on the correct non-zero entry.

The L sections of the SPARC are divided into B *blocks* of L/B sections each. Each section within a block is allocated the same power. For example, with $L = 512$ and $B = 32$, there are 32 blocks with 16 sections per block. The algorithm sequentially allocates power to each of the B blocks as follows. Allocate the minimum power to the first block of sections so that they can be decoded in the first iteration when $\tau_0^2 = \sigma^2 + P$. Using (4.3), we set the power in each section of the first block to

$$P_\ell = \frac{2R\tau_0^2}{L}, \quad 1 \leq \ell \leq \frac{L}{B}.$$

Using (4.3), we estimate $x_1 = x(\tau_0) = BP_1$, and hence $\tau_1^2 = \sigma^2 + (P - BP_1)$. Using this value, allocate the minimum required power for the second block of sections to decode, i.e., $P_\ell = 2R\tau_1^2/L$ for $\frac{L}{B} + 1 \leq \ell \leq \frac{2L}{B}$. If we sequentially allocate power in this manner to each of the B blocks, then the total power allocated by this scheme will be strictly less than P whenever $R < \mathcal{C}$. We therefore modify the scheme as follows.

For $1 \leq b \leq B$, to allocate power to the bth block of sections assuming that the first $(b - 1)$ blocks have been allocated, we compare the two options and choose the one that allocates higher power to the block: i) allocating the minimum required power (computed as above) for the bth block
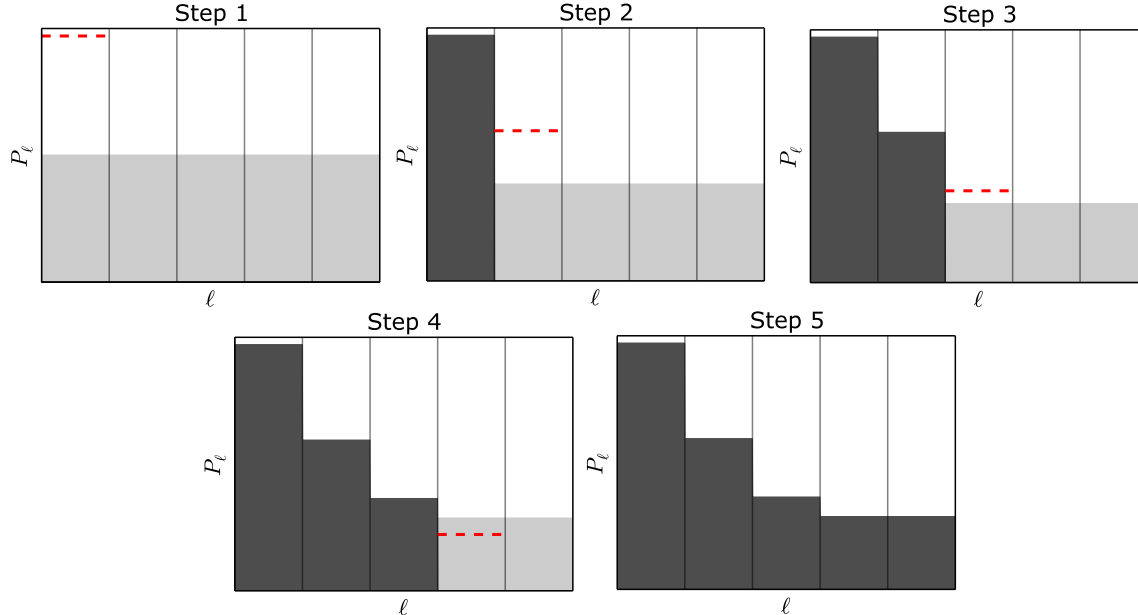
61

Figure 4.2: Example illustrating the iterative power allocation algorithm with $B = 5$. In each step, the height of the light gray region represents the allocation that distributes the remaining power equally over all the remaining sections. The dashed red line indicates the minimum power required for decoding the current block of sections. The dark gray bars represent the power that has been allocated at the beginning of the current step.

of sections to decode; ii) allocating the remaining available power equally to sections in blocks $b, \ldots, B$, and terminating the algorithm. This gives a flattening in the final blocks similar to the allocation in (4.2), but without requiring a specific parameter that determines where the flattening begins. The iterative power allocation routine is described in Algorithm 1. Figure 4.2 shows a toy example building up the power allocation for $B = 5$, where flattening is seen to occur in step 4.

Choosing B: By construction, the iterative power allocation scheme specifies the number of iterations of the AMP decoder in the large system limit. This is given by the number of blocks with distinct powers; in particular the number of iterations (in the large system limit) is of the order of B. For finite code lengths, we find that it is better to use the termination criterion described in 4.1.2 based on the online estimates $\widehat{\tau}_t^2$. This termination criterion allows us to choose the number of blocks B to be as large as L. We found that choosing $B = L$, together with the termination criterion consistently gives a small improvement in error performance (compared to other choices of B), with no additional time or memory cost.

Additionally, with $B = L$, it is possible to quickly determine a pair (a, f) for the modified exponential allocation in (4.2) which gives a nearly identical allocation to the iterative algorithm. This is done by first setting f to obtain the same flattening point found in the iterative allocation, and then searching for an a which matches the first allocation coefficient P_1 between the iterative and the modified exponential allocations. Consequently, any simulation results obtained for the iterative power allocation could also be obtained using a suitable (a, f) with the modified exponential allocation, without having to first perform a costly numerical optimization over (a, f).

Algorithm 1 Iterative power allocation routine

Require: L, B, σ^2, P, R such that B divides L.
  Initialise $k \leftarrow \frac{L}{B}$
  for $b = 0$ to $B - 1$ **do**
    $P_{\text{remain}} \leftarrow P - \sum_{\ell=1}^{bk} P_\ell$
    $\tau^2 \leftarrow \sigma^2 + P_{\text{remain}}$
    $P_{\text{block}} \leftarrow 2R\tau^2/L$
    if $P_{\text{remain}}/(L - bk) > P_{\text{block}}$ **then**
      $P_{bk+1}, \ldots, P_L \leftarrow P_{\text{remain}}/(L - bk)$
      break
    else
      $P_{bk+1}, \ldots, P_{(b+1)k} \leftarrow P_{\text{block}}$
    end if
  end for
  return $P_1, \ldots, P_L$

Figure 4.3 compares the error performance of the exponential and iterative power allocation schemes discussed above for different values of R at $\mathsf{snr} = 7, 15, 31$. Compared to the original exponential power allocation, the iterative allocation has significantly improved error performance for rates away from capacity. It also generally outperforms the modified exponential allocation results, as seen Figure 4.1, where the bottom curve (green) shows the error performance of the iterative allocation.

For the experiments in Figure 4.3, the value for R used in constructing the iterative allocation (denoted by R_{PA}) was optimized numerically. Constructing an iterative allocation with $R = R_{PA}$ yields good results, but due to finite length concentration effects, the R_{PA} yielding the smallest average error rate may be slightly different from the communication rate R. The effect of R_{PA} on the concentration of error rates is discussed in Section 4.3.2. We emphasize that this optimization over R_{PA} is simpler than numerically optimizing the pair (a, f) for the modified exponential allocation. Furthermore, guidelines for choosing R_{PA} as a function of R are given in Section 4.3.2.

4.3   Code parameter choices at finite code lengths

In this section, we discuss how the choice of SPARC design parameters can influence finite length error performance with the AMP decoder. We will see that the parameters (L, M) and the power allocation both inducee a trade-off between the 'typical' value of section error rate predicted by state evolution, and concentration of actual error rates around the typical values.

If the termination step is T, then we expect the test statistic in the final iteration to be $\mathsf{stat}^T \approx \beta + \tau_T Z$, where τ_T is determined from the SE equations. (For reliable decoding, we expect τ_T^2 to be close to σ^2.) This leads to the following SE-based prediction for the section error rate [51, Proposition 1]:

$$\bar{\mathcal{E}}_{\text{sec}} = 1 - \frac{1}{L} \sum_{\ell=1}^{L} \mathbb{E}_U \left[\Phi \left(\frac{\sqrt{nP_\ell}}{\sigma} + U \right) \right]^{M-1}. \tag{4.4}$$
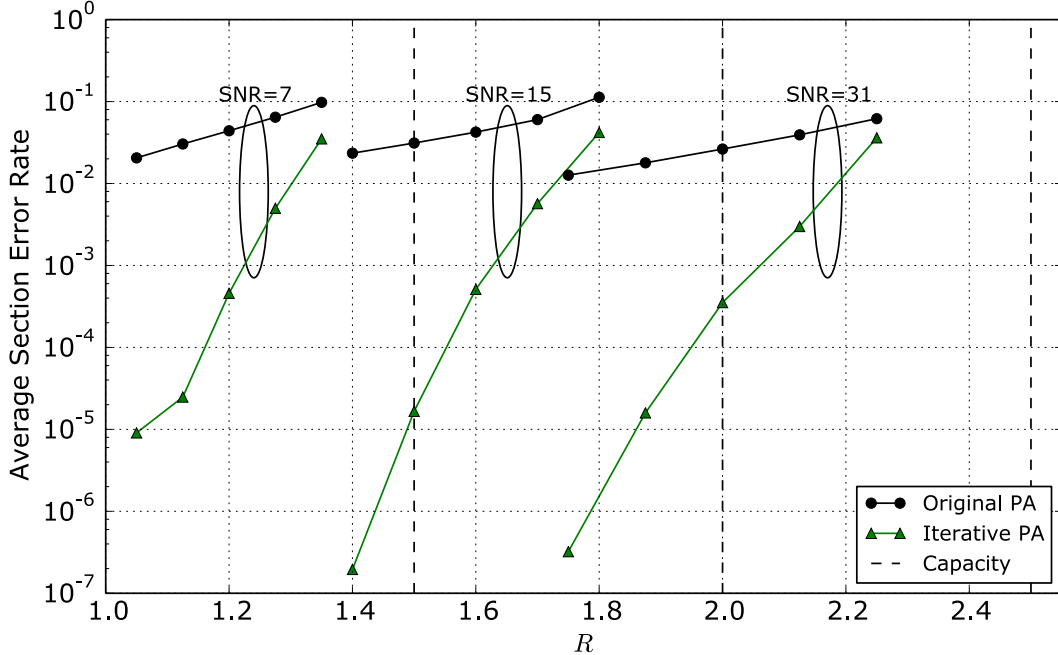
Figure 4.3: AMP section error rate vs R (in bits) at $\mathsf{snr} = 7, 15, 31$, corresponding to $\mathcal{C} = 1.5, 2, 2.5$ bits (shown with dashed vertical lines). At each snr, the section error rate is reported for rates $R/\mathcal{C} = 0.70, 0.75, 0.80, 0.85, 0.90$. The SPARC parameters are $M = 512, L = 1024$. The top black curve shows the error rate with the exponential allocation $P_\ell \propto 2^{-2\mathcal{C}\ell/L}$ (with $\mathcal{C}$ in bits). The lower green curve shows the error rate with iterative power allocation, with $B = L$.

4.3.1 Effect of L and M on concentration

To understand the effect of increasing M, consider Figure 4.4 which shows the error performance of a SPARC with $R = 1.5, L = 1024$, as we increase the value of M. Since $n = L \log M / R$, the code length n increases logarithmically with M for a fixed L. We observe that the section error rate (averaged over 200 trials) decreases with M up to $M = 2^9$, and then starts increasing. This is in sharp contrast to the SE prediction (4.4) (dashed line in Figure 4.4) which keeps decreasing as M is increased.

This divergence between the actual section error rate and the SE prediction for large M is due to large fluctuations in the number of section errors across trials. Theorem 3.3 shows how the concentration of section error rates near the SE prediction depends on L and M. Since the probability bound in (3.72) depends on the ratio $L/(\log M)^{2T-1}$, for a given L the probability of large deviations from the SE prediction increases with M.

This leads to the situation shown in Figure 4.4, which shows that the SE prediction $\mathcal{E}_{\mathrm{sec}}^{\mathrm{SE}}$ continues to decrease with M, but beyond a certain value of M, the observed average section error rate becomes progressively worse due to loss of concentration. This is caused by a small number of trials with a very large number of section errors, even as the majority of trials experience lower and lower error rates as M is increased. This effect can be clearly seen in Figure 4.5, which compares the histogram of section error rates over 200 trials for $M = 64$ and $M = 4096$. The distribution of
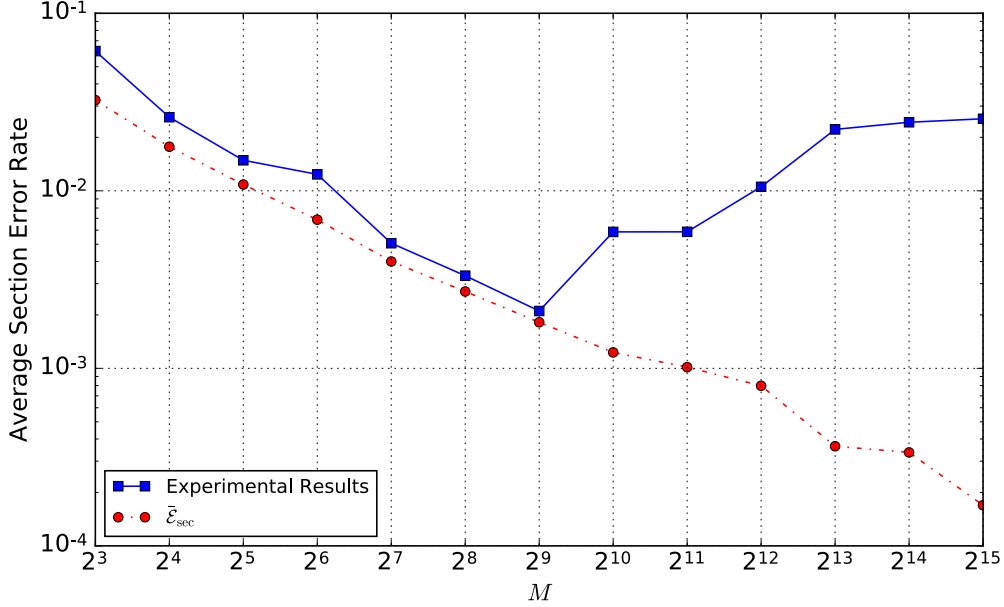
64

Figure 4.4: AMP error performance with increasing M, for $L = 1024$, $R = 1.5$ bits, and $\frac{E_b}{N_0} = 5.7$ dB (2 dB from Shannon limit).

errors is clearly different, but both cases have the same average section error rate due to the poorer concentration for $M = 4096$.

Therefore, given R, snr, and L, there is an optimal M that minimizes the empirical section error rate. Beyond this value of M, the benefit from any further increase is outweighed by the loss of concentration. For a given R, values of M close to L are a good starting point for optimizing the empirical section error rate, but obtaining closed-form estimates of the optimal M for a given L is still an open question.

4.3.2 Effect of power allocation on concentration

The non-asymptotic result of Lemma 3.6 indicates that at finite lengths, the minimum power required for a section ℓ to decode in an iteration may be slightly different than that indicated by the asymptotic approximation in (4.3). Recall that the iterative power allocation algorithm in Section 4.2.1 was designed based on (4.3). We can compensate for the difference between the approximation and the actual value of $x(\tau)$ by running the iterative power allocation in Algorithm 1 using a modified rate R_{PA} which may be slightly different from the communication rate R.

If we run the power allocation algorithm with $R_{\mathrm{PA}} > R$, from (4.3) we see that additional power is allocated to the initial blocks, at the cost of less power for the final blocks (where the allocation is flat). Conversely, choosing $R_{\mathrm{PA}} < R$ allocates less power to the initial blocks, and increases the power in the final sections which have a flat allocation. This increases the likelihood of the initial section being decoded in error; in a trial when this happens, there will be a large number of section errors. However, if the initial sections are decoded correctly, the additional power in the final sections increases the probability of the trial being completely error-free. Thus choosing
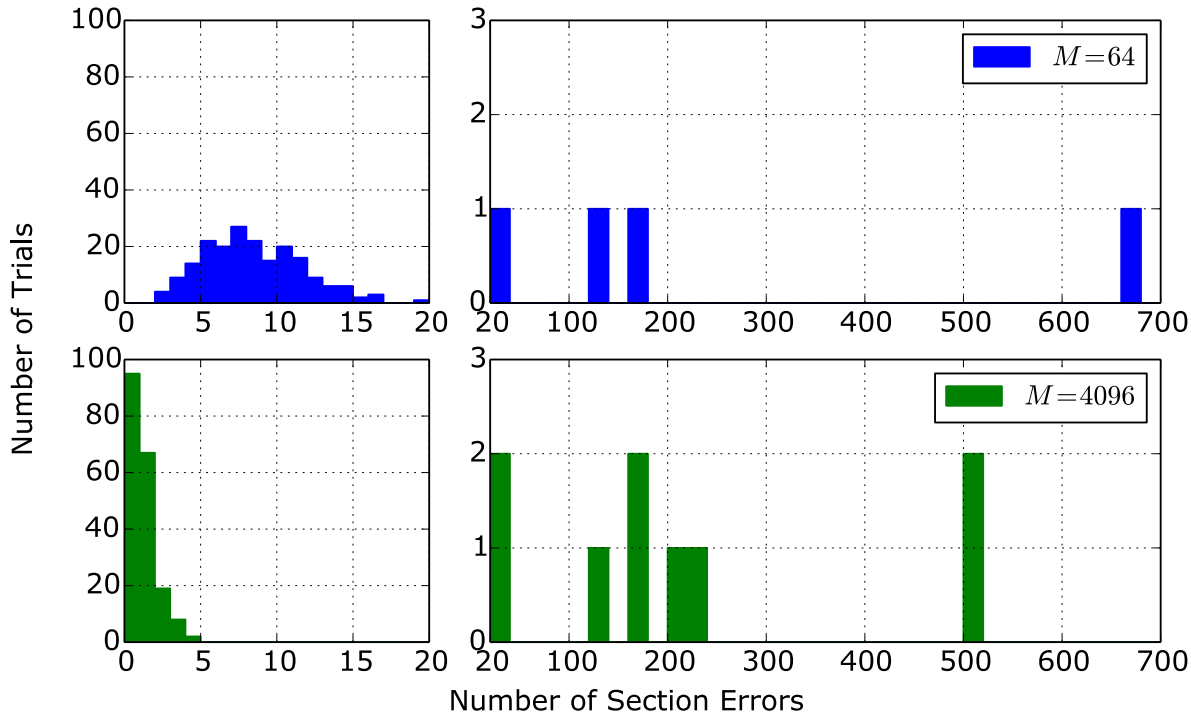
65

Figure 4.5: Histogram of AMP section errors over 200 trials $M = 64$ (top) and $M = 4096$ (bottom), with $L = 1024$, $R = 1.5$ bits, $\frac{E_b}{N_0} = 5.7$dB. The left panels highlight distribution of errors around low section error counts, while the right panels show the distribution around high-error-count events. As shown in Figure 4.4, both cases have an average section error rate of around 10^{-2}.

$R_{PA} < R$ makes completely error-free trials more likely, but also increases the likelihood of having trials with a large number of sections in error.

To summarize, the larger the R_{PA}, the better the concentration of section error rates of individual trials around the overall average. However, increasing R_{PA} beyond a point just increases the average section error rate because of too little power being allocated to the final sections.

Through numerical experiments, we found that the value of $\frac{R_{PA}}{R}$ that minimizes the average section error rate increases with R. In particular, the optimal $\frac{R_{PA}}{R}$ was 0 for $R \leq 1$ bit; the optimal $\frac{R_{PA}}{R}$ for $R = 1.5$ bits was close to 1, and for $R = 2$ bits, the optimal $\frac{R_{PA}}{R}$ was between 1.05 and 1.1. Though this provides a useful design guideline, a deeper theoretical analysis of the role of R_{PA} in optimizing the finite length error performance is an open question.

4.4 Comparison with coded modulation

We compare the error performance of AMP-decoded SPARCs with coded modulation with LDPC codes from the WiMax standard IEEE 802.16e. For the latter, we consider: 1) A 16-QAM constellation with a rate $\frac{1}{2}$ LDPC code for an overall rate $R = 1$ bit/channel use/real dimension,
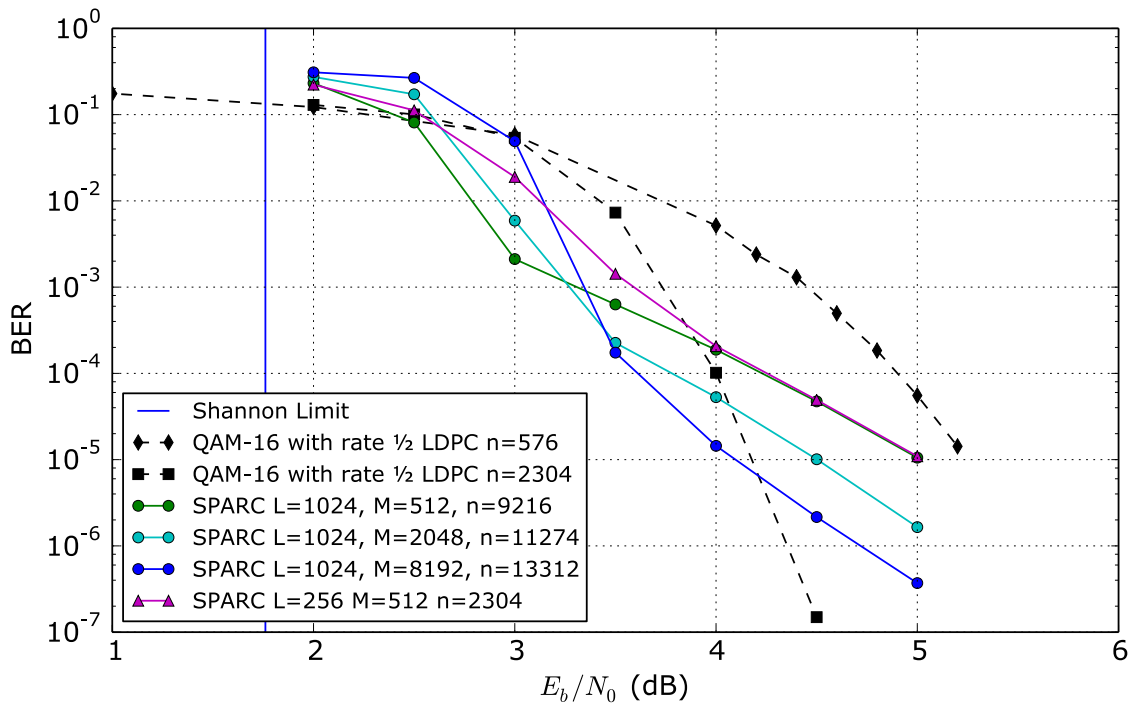
66

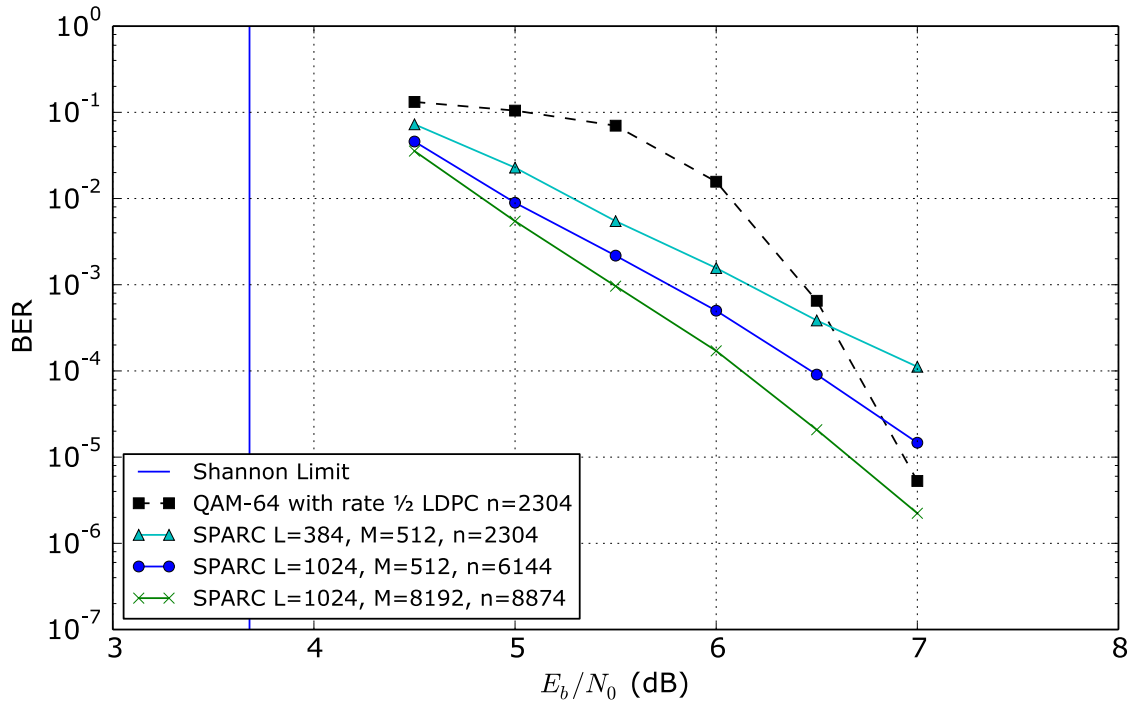Figure 4.6: Comparison with LDPC coded modulation at $R = 1$ bit

Figure 4.7: Comparison with LDPC coded modulation at $R = 1.5$ bits

67

and 2) A 64-QAM constellation with a rate $\frac{1}{2}$ LDPC code for an overall rate $R = 1.5$ bits/channel use/real dimension. (The spectral efficiency is $2R$ bits/s/Hz.) The coded modulation results, shown in dashed lines in Figures 4.6 and 4.7, are obtained using the CML toolkit [2] with LDPC code lengths $n = 576$ and $n = 2304$.

Throughout this section and the next, rate will be measured in *bits*.

Each figure compares the bit error rates (BER) of the coded modulation schemes with various SPARCs of the same rate, including a SPARC with a matching code length of $n = 2304$. Using $P = E_b R$ and $\sigma^2 = \frac{N_0}{2}$, the signal-to-noise ratio of the SPARC can be expressed as $\frac{P}{\sigma^2} = \frac{2RE_b}{N_0}$. The SPARCs are implemented using Hadamard-based design matrices, power allocation designed using the iterative algorithm in Sec. 4.2.1 with $B = L$, parameters $\widehat{\tau}_t^2$ computed online, and the early termination criterion described in 4.1.2. A Jupyter notebook detailing the SPARC implementation in Python is available at [1].

4.5   AMP with partial outer codes

Figures 4.6 and 4.7 show that for block lengths of the order of a few thousands, AMP-decoded SPARCs do not exhibit a steep waterfall in section error rate. Even at high E_b/N_0 values, it is still common to observe a small number of section errors. If these could be corrected, we could hope to obtain a sharp waterfall behavior similar to the LDPC codes.

In the simulations of the AMP decoder described above, when M and R_{PA} are chosen such that the average error rates are well-concentrated around the state evolution prediction, the number of section errors observed is similar across trials. Furthermore, we observe that the majority of sections decoded incorrectly are those in the flat region of the power allocation, i.e., those with the lowest allocated power. This suggests we could use a high-rate outer code to protect just these sections, sacrificing some rate, but less than if we naïvely protected all sections. We call the sections covered by the outer code *protected* sections, and conversely the earlier sections which are not covered by the outer code are *unprotected*. In [16], it was shown that a Reed-Solomon outer code (that covered all the sections) could be used to obtain a bound the probability of codeword error from a bound on the probability of excess section error rate.

Encoding with an outer code (e.g., LDPC or Reed-Solomon code) is straightforward: just replace the message bits corresponding to the protected sections with coded bits generated using the usual encoder for the chosen outer code. To decode, we would like to obtain bit-wise posterior probabilities for each codeword bit of the outer code, and use them as inputs to a soft-information decoder, such as a sum-product or min-sum decoder for LDPC codes. The output of the AMP decoding algorithm permits this: it yields β^T, which contains weighted *section-wise* posterior probabilities; we can directly transform these into *bit-wise* posterior probabilities. See Algorithm 2 for details.

Moreover, in addition to correcting AMP decoding errors in the protected sections, successfully decoding the outer code also provides a way to correct remaining errors in the unprotected sections of the SPARC codeword. Indeed, after decoding the outer code we can subtract the contribution of the protected sections from the channel output sequence y, and re-run the AMP decoder on just
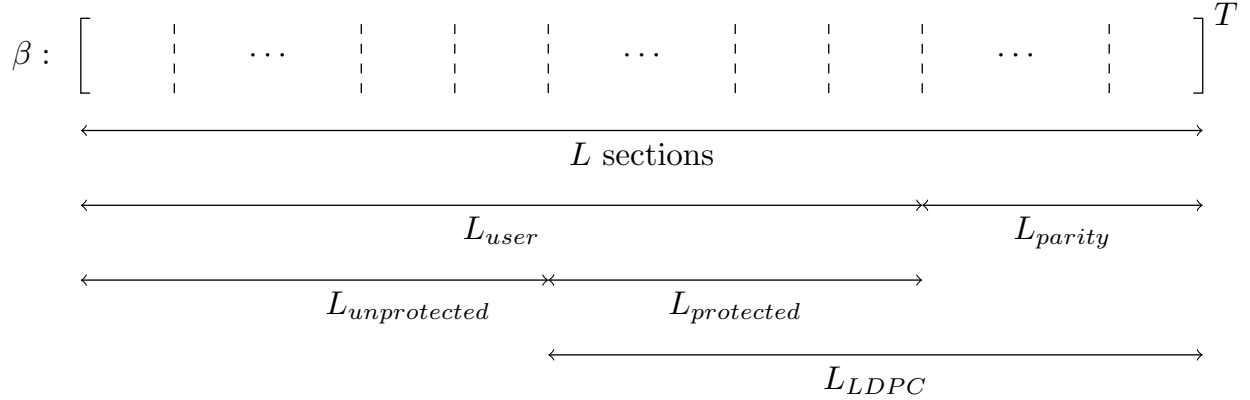
Figure 4.8: Division of the L sections of β for an outer LDPC code

the unprotected sections. The key point is that subtracting the contribution of the later (protected) sections eliminates the interference due to these sections; then running the AMP decoder on the unprotected sections is akin to operating at a much lower rate.

Thus the decoding procedure has three stages: i) first round of AMP decoding, ii) decoding the outer code using soft outputs from the AMP, and iii) subtracting the contribution of the sections protected by the outer code, and running the AMP decoder again for the unprotected sections. We find that the final stage, i.e., running the AMP decoder again after the outer code recovers errors in the protected sections of the SPARC, provides a significant advantage over a standard application of an outer code, i.e., decoding the final codeword after the second stage.

We describe this combination of SPARCs with outer codes below, using an LDPC outer code. The resulting error rate curves exhibit sharp waterfalls in final error rates, even when the LDPC code only covers a minority of the SPARC sections.

We use a binary LDPC outer code with rate R_{LDPC}, block length n_{LDPC} and code dimension k_{LDPC}, so that $k_{LDPC}/n_{LDPC} = R_{LDPC}$. For clarity of exposition we assume that both n_{LDPC} and k_{LDPC} are multiples of $\log M$ (and consequently that M is a power of two). As each section of the SPARC corresponds to $\log M$ bits, if $\log M$ is an integer, then n_{LDPC} and k_{LDPC} bits represent an integer number of SPARC sections, denoted by

$$L_{LDPC} = \frac{n_{LDPC}}{\log M} \quad \text{and} \quad L_{protected} = \frac{k_{LDPC}}{\log M},$$

respectively. The assumption that k_{LDPC} and n_{LDPC} are multiples of $\log M$ is not necessary in practice; the general case is discussed at the end of the next subsection.

We partition the L sections of the SPARC codeword as shown in Fig 4.8. There are L_{user} sections corresponding to the user (information) bits; these sections are divided into *unprotected* and *protected* sections, with only the latter being covered by the outer LDPC code. The parity bits of the LDPC codeword index the last L_{parity} sections of the SPARC. For convenience, the *protected* sections and the *parity* sections together are referred to as the *LDPC* sections.

For a numerical example, consider the case where $L = 1024$, $M = 256$. There are $\log M = 8$ bits per SPARC section. For a $(5120, 4096)$ LDPC code ($R_{LDPC} = 4/5$) we obtain the following

69

Algorithm 2 Weighted position posteriors β_ℓ to bit posteriors $p_0, \ldots, p_{\log M - 1}$ for section $\ell \in [L]$

Require: $\beta_\ell = [\beta_{\ell,1}, \ldots, \beta_{\ell,M}]$, for M a power of 2
  Initialise bit posteriors $p_0, \ldots, p_{\log M - 1} \leftarrow 0$
  Initialise normalization constant $c \leftarrow \sum_{i=1}^{M} \beta_{\ell,i}$
  for $\log i = 0, 1, \ldots, \log M - 1$ **do**
    $b \leftarrow \log M - \log i - 1$
    $k \leftarrow i$
    while $k < M$ **do**
      for $j = k + 1, k + 2, \ldots, k + i$ **do**
        $p_b \leftarrow p_b + \beta_{\ell,j}/c$
      end for
      $k \leftarrow k + 2i$
    end while
  end for
  return $p_0, \ldots, p_{\log M - 1}$

---

relationships between the number of the sections of each kind:

$$L_{parity} = \frac{n_{LDPC} - k_{LDPC}}{\log M} = \frac{(5120 - 4096)}{8} = 128,$$

$$L_{user} = L - L_{parity} = 1024 - 128 = 896,$$

$$L_{protected} = \frac{k_{LDPC}}{\log M} = \frac{4096}{8} = 512,$$

$$L_{LDPC} = L_{protected} + L_{parity} = 512 + 128 = 640,$$

$$L_{unprotected} = L_{user} - L_{protected} = L - L_{LDPC} = 384.$$

There are $L_{user} \log M = 7168$ user bits, of which the final $k_{LDPC} = 4096$ are encoded to a systematic $n_{LDPC} = 5120$-bit LDPC codeword. The resulting $L \log M = 8192$ bits (including both the user bits and the LDPC parity bits) are encoded to a SPARC codeword using the SPARC encoder and power allocation described in previous sections.

We continue to use R to denote the overall user rate, and n to denote the SPARC code length so that $nR = L_{user} \log M$. The underlying SPARC rate (including the overhead due to the outer code) is denoted by R_{SPARC}. We note that $nR_{SPARC} = L \log M$, hence $R_{SPARC} > R$. For example, with $R = 1$ and L, M and the outer code parameters as chosen above, $n = L_{user}(\log M)/R = 7168$, so $R_{SPARC} = 1.143$.

4.5.1 Decoding SPARCs with LDPC outer codes

At the receiver, we decode as follows:

1. Run the AMP decoder to obtain β^T. Recall that entry j within section ℓ of β^T is proportional to the posterior probability of the column j being the transmitted one for section ℓ. Thus the AMP decoder gives section-wise posterior probabilities for each section $\ell \in [L]$.

2. Convert the section-wise posterior probabilities to bit-wise posterior probabilities using Algorithm 2, for each of the L_{LDPC} sections. This requires $O(L_{LDPC} M \log M)$ time complexity, of the same order as one iteration of AMP.

3. Run the LDPC decoder using the bit-wise posterior probabilities obtained in Step 2 as inputs.

4. If the LDPC decoder fails to produce a valid LDPC codeword, terminate decoding here, using β^T to produce $\hat{\beta}$ by selecting the maximum value in each section (as per usual AMP decoding).

5. If the LDPC decoder succeeds in finding a valid codeword, we use it to re-run AMP decoding on the unprotected sections. For this, first convert the LDPC codeword bits to a partial $\hat{\beta}_{LDPC}$ as follows, using a method similar to the original SPARC encoding:

   5.1 Set the first $L_{unprotected} M$ entries of $\hat{\beta}_{LDPC}$ to zero,

   5.2 The remaining L_{LDPC} sections (with M entries per section) of $\hat{\beta}_{LDPC}$ will have exactly one-non zero entry per section, with the LDPC codeword determining the location of the non-zero in each section. Indeed, noting that $n_{LDPC} = L_{LDPC} \log M$, we consider the LDPC codeword as a concatenation of L_{LDPC} blocks of $\log M$ bits each, so that each block of bits indexes the location of the non-zero entry in one section of $\hat{\beta}_{LDPC}$. The value of the non-zero in section ℓ is set to $\sqrt{nP_\ell}$, as per the power allocation.

   Now subtract the codeword corresponding to $\hat{\beta}_{LDPC}$ from the original channel output y, to obtain $y' = y - A\hat{\beta}_{LDPC}$.

6. Run the AMP decoder again, with input y', and operating only over the first $L_{unprotected}$ sections. As this operation is effectively at a much lower rate than the first decoder (since the interference contribution from all the protected sections is removed), it is more likely that the unprotected bits are decoded correctly than in the first AMP decoder.

   We note that instead of generating y', one could run the AMP decoder directly on y, but enforcing that in each AMP iteration, each of the L_{LDPC} sections has all its non-zero mass on the entry determined by $\hat{\beta}_{LDPC}$, i.e., consistent with Step 5.b).

7. Finish decoding, using the output of the final AMP decoder to find the first $L_{unprotected} M$ elements of $\hat{\beta}$, and using $\hat{\beta}_{LDPC}$ for the remaining $L_{LDPC} M$ elements.

4.5.2 Simulation results

The combined AMP and outer LDPC setup described above was simulated using the (5120, 4096) LDPC code ($R_{LDPC} = 4/5$) specified in [24] with a min-sum decoder. Bit error rates were measured only over the user bits, ignoring any bit errors in the LDPC parity bits.

Figure 4.9 plots results at overall rate $R = \frac{4}{5}$, where the underlying LDPC code (modulated with BPSK) can be compared to the SPARC with LDPC outer code, and to a plain SPARC with rate $\frac{4}{5}$. In this case $R_{PA} = 0$, giving a flat power allocation. Figure 4.10 plots results at overall rate $R = 1.5$, where we can compare to the QAM-64 WiMAX LDPC code, and to the plain SPARC with rate 1.5 of Figure 4.7.
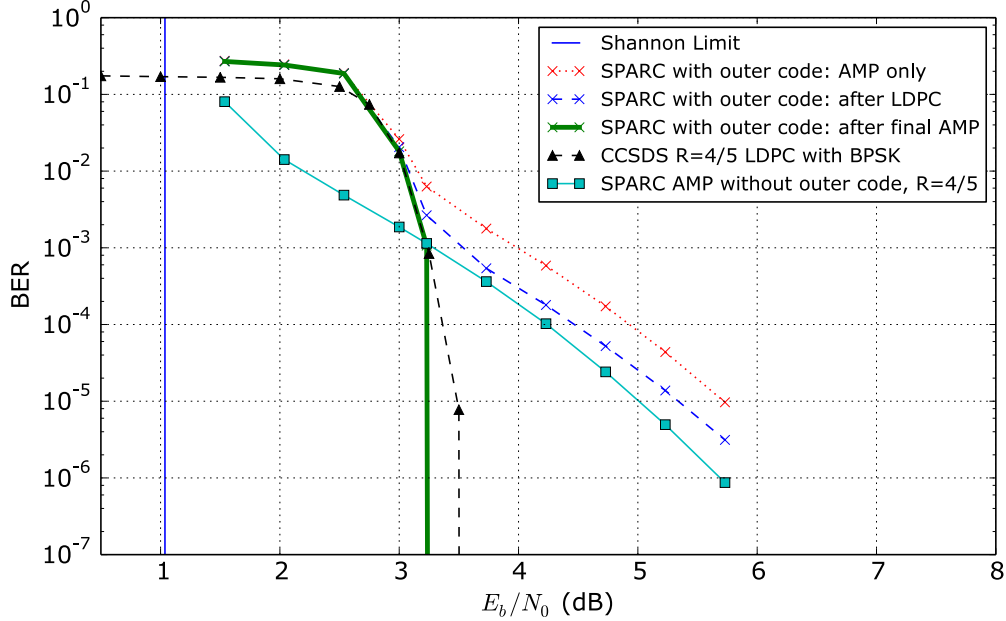
Figure 4.9: Comparison to plain AMP and to BPSK-modulated LDPC at overall rate $R = 0.8$. The SPARCs are both $L = 768$, $M = 512$. The underlying SPARC rate when the outer code is included is $R_{SPARC} = 0.94$. The BPSK-modulated LDPC is the same CCSDS LDPC code [24] used for the outer code. For this configuration, $L_{user} = 654.2$, $L_{parity} = 113.8$, $L_{unprotected} = 199.1$, $L_{protected} = 455.1$, and $L_{LDPC} = 568.9$.

The plots show that protecting a fraction of sections with an outer code does provide a steep waterfall above a threshold value of $\frac{E_b}{N_0}$. Below this threshold, the combined SPARC + outer code has worse error performance than the plain rate R SPARC without the outer code. This can be explained as follows. The combined code has a higher SPARC rate $R_{SPARC} > R$, which leads to a larger section error rate for the first AMP decoder, and consequently, to worse bit-wise posteriors at the input of the LDPC decoder. For $\frac{E_b}{N_0}$ below the threshold, the noise level at the input of the LDPC decoder is beyond than the error-correcting capability of the LDPC code, so the LDPC code effectively does not correct any section errors. Therefore the overall error rate is worse with the outer code.

Above the threshold, we observe that the second AMP decoder (after subtracting the contribution of the LDPC-protected sections) is successful at decoding the unprotected sections that were initially decoded incorrectly. This is especially apparent in the $R = \frac{4}{5}$ case (Figure 4.9), where the section errors are uniformly distributed over all sections due to the flat power allocation; errors are just as likely in the unprotected sections as in the protected sections.

4.5.3   Outer code design choices

The error performance with an outer code is sensitive to what fraction of sections are protected by the outer code. When more sections are protected by the outer code, the overhead of using the outer code is also higher, driving R_{SPARC} higher for the same overall user rate R. This leads
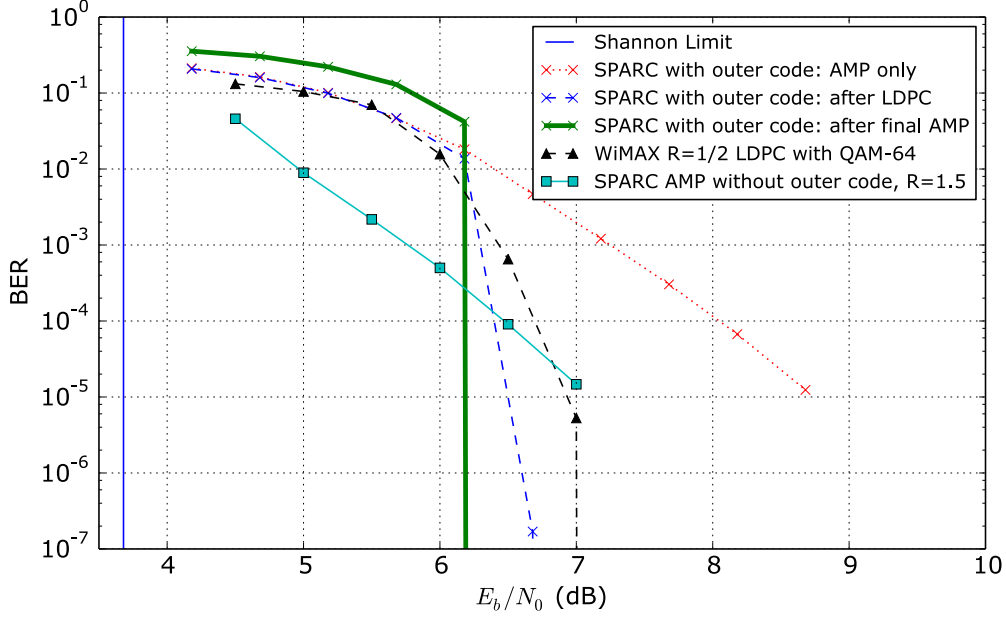
Figure 4.10: Comparison to plain AMP and to the QAM-64 WiMAX LDPC of Section 4.4 at overall rate $R = 1.5$ The SPARCs are both $L = 1024$, $M = 512$. The underlying SPARC rate including the outer code is $R_{SPARC} = 1.69$. For this configuration, $L_{user} = 910.2$, $L_{parity} = 113.8$, $L_{unprotected} = 455.1$, $L_{protected} = 455.1$, and $L_{LDPC} = 455.1$.

to worse error performance in the initial AMP decoder, which has to operate at the higher rate R_{SPARC}. As discussed above, if R_{SPARC} is increased too much, the bit-wise posteriors input to the LDPC decoder are degraded beyond its ability to successfully decode, giving poor overall error rates.

Since the number of sections covered by the outer code depends on both $\log M$ and n_{LDPC}, various trade-offs are possible. For example, given n_{LDPC}, choosing a larger value of $\log M$ corresponds to fewer sections being covered by the outer code. This results in smaller rate overhead, but increasing $\log M$ may also affect concentration of the error rates around the SE predictions, as discussed in Section 4.3. We conclude with two remarks about the choice of parameters for the SPARC and the outer code.

1. When using an outer code, it is highly beneficial to have good concentration of the section error rates for the initial AMP decoder. This is because a small number of errors in a single trial can usually be fully corrected by the outer code, while occasional trials with a very large number of errors cannot.

2. Due to the second AMP decoder operation, it is not necessary for all sections with low power to be protected by the outer code. For example, in Figure 4.9, all sections have equal power, and around 30% are not protected by the outer code. Consequently, these sections are often not decoded correctly by the first decoder. Only once the protected sections are removed is the second decoder able to correctly decode these unprotected sections. In general the aim should be to cover all or most of the sections in the flat region of the power allocation, but experimentation is necessary to determine the best trade-off.

73

An interesting direction for future work would be to develop an EXIT chart analysis [107, 6, 93] to jointly optimize the design of the SPARC and the outer LDPC code.

Chapter 5

Spatially Coupled SPARCs

The efficient capacity-achieving decoders discussed in Chapter 3 all relied on power allocation across the sections. The design matrix was chosen with independent, identically distributed Gaussian entries, while the values of the non-zero coefficients varied across sections of the codeword. Equivalently, one can define the power allocation by changing the variance of the Gaussian entries in each section of the design matrix, while the non-zero coefficients of the codeword all have the same value. *Spatial coupling* is a generalization of the latter view of power allocation.

In a spatially coupled SPARC (SC-SPARC), the design matrix is divided into multiple blocks, each with independent zero-mean Gaussian entries of a specified variance. The variance of the entries may vary across blocks, while the values of the non-zero entries in the message vector are all equal. Within this general framework, we will consider a simple construction with a band-diagonal spatially coupled design matrix, and show that it can achieve the AWGN capacity with AMP decoding without the need for power allocation. Furthermore, at finite code lengths, numerical simulations indicate that SC-SPARCs have a much steeper decay of error rate than power allocated SPARCs as we as we back off from the Shannon limit.

The idea of spatial coupling was introduced in the context of LDPC codes [43, 78, 75, 74, 82], and later applied to compressed sensing in [73, 72, 35]. Subsequently, spatially coupled SPARCs were studied by Barbier et al. in [14, 8, 9, 12, 10]. The discussion in this chapter is largely based on the spatially coupled SPARC construction and analysis presented in [59, 96].

5.1   Spatially coupled SPARC construction

As in the standard construction, a spatially coupled (SC) SPARC is defined by a design matrix A of dimension $n \times ML$, where n is the code length. The codeword is $x = A\beta$, where β has one non-zero entry in each of the L sections.

In an SC-SPARC, the matrix A consists of independent zero-mean normally distributed entries whose variances are specified by a *base matrix* W of dimension $L_R \times L_C$. The design matrix A
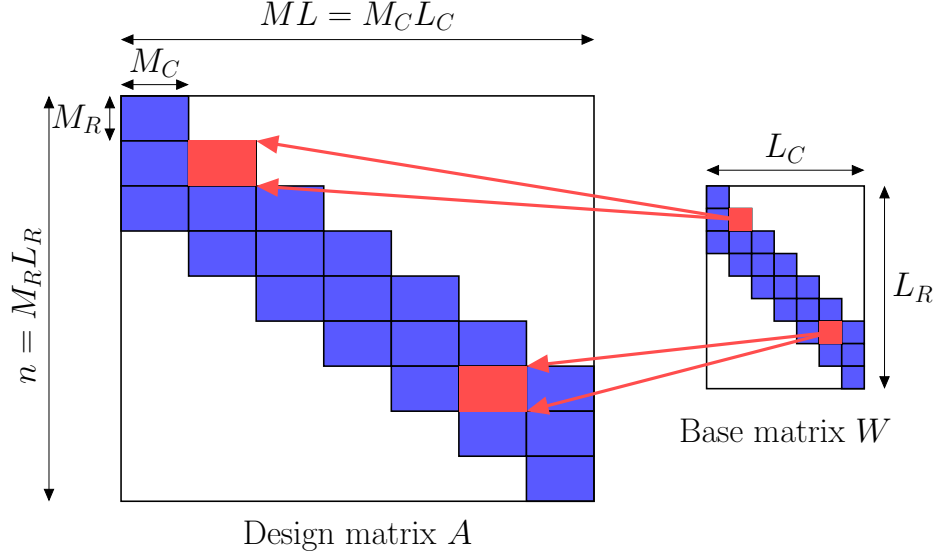
Figure 5.1: A spatially coupled design matrix A is divided into blocks of size $M_R \times M_C$. There are L_R and L_C blocks in each column and row respectively. The independent matrix entries are normally distributed, $A_{ij} \sim \mathcal{N}(0, \frac{1}{L} W_{\mathsf{r}(i)\mathsf{c}(j)})$, where W is the base matrix. The base matrix shown here is an (ω, Λ) base matrix with parameters $\omega = 3$ and $\Lambda = 7$. The white parts of A and W correspond to zeros.

is obtained from the base matrix W by replacing each entry W_{rc}, for $r \in [L_R]$, $c \in [L_C]$, by an $M_R \times M_C$ block with i.i.d. entries $\sim \mathcal{N}(0, W_{rc}/L)$. This is analogous to the "graph lifting" procedure in constructing SC-LDPC codes from protographs [82]. See Fig. 5.1 for an example, and note that $n = L_R M_R$ and $ML = L_C M_C$.

From the construction, the design matrix has independent normal entries

$$A_{ij} \sim \mathcal{N}\left(0, \frac{1}{L} W_{\mathsf{r}(i)\mathsf{c}(j)}\right) \ \forall \ i \in [n], \ j \in [ML]. \tag{5.1}$$

The operators $\mathsf{r}(\cdot) : [n] \to [L_R]$ and $\mathsf{c}(\cdot) : [ML] \to [L_C]$ in (5.1) map a particular row or column index to its corresponding *row block* or *column block* index. We require L_C to divide L, resulting in $\frac{L}{L_C}$ sections per column block.

The non-zero coefficients of β are all set to 1. Then, to satisfy the power constraint $\frac{1}{n}\|x\|^2 = P$, it can be shown that the entries of the base matrix W must satisfy

$$\frac{1}{L_R L_C} \sum_{r=1}^{L_R} \sum_{c=1}^{L_C} W_{rc} = P. \tag{5.2}$$

The trivial base matrix with $L_R = L_C = 1$ corresponds to a standard (non-SC) SPARC with flat power allocation (discussed in Chapter 2), while a single-row base matrix $L_R = 1$, $L_C = L$ is equivalent to a standard SPARC with power allocation (Chapters 3 and 4). Without loss of generality, we will assume that $\frac{1}{L_C} \sum_{c=1}^{L_C} W_{rc}$ and $\frac{1}{L_R} \sum_{r=1}^{L_R} W_{rc}$ are bounded above and below by strictly positive constants.

Here, we will use the following base matrix inspired by the coupling structure of SC-LDPC codes constructed from protographs [82].

Definition 5.1. *An (ω, Λ) base matrix W for SC-SPARCs is described by two parameters: coupling width $\omega \geq 1$ and coupling length $\Lambda \geq 2\omega - 1$. The matrix has $L_R = \Lambda + \omega - 1$ rows, $L_C = \Lambda$ columns, with each column having ω identical non-zero entries. For an average power constraint P, the (r, c)th entry of the base matrix, for $r \in [L_R], c \in [L_C]$, is given by*

$$
W_{rc} = \begin{cases} P \cdot \frac{\Lambda + \omega - 1}{\omega} & \text{if } c \leq r \leq c + \omega - 1, \\ 0 & \text{otherwise.} \end{cases} \tag{5.3}
$$

For example, the base matrix in Fig. 5.1 has parameters $\omega = 3$ and $\Lambda = 7$. This base matrix construction was also used in [79] for SC-SPARCs. Other base matrix constructions can be found in [72, 35, 8, 12].

Each non-zero entry in an (ω, Λ) base matrix W corresponds to an $M_R \times (ML/L_C)$ block in the design matrix A. Each of these blocks can be viewed as a standard (non-SC) SPARC with $\frac{L}{L_C}$ sections (with M columns in each section), code length M_R, and rate $R_{\text{inner}} = \frac{(L/L_C) \ln M}{M_R}$ nats. Since $nR = L \ln M$, the overall rate of the SC-SPARC is related to R_{inner} according to

$$
R = \frac{\Lambda}{\Lambda + \omega - 1} R_{\text{inner}}. \tag{5.4}
$$

The coupling width ω is usually an integer greater than 1, so $R < R_{\text{inner}}$. The difference $(R_{\text{inner}} - R)$ is often referred to as a rate loss. The rate loss depends on the ratio $(\omega - 1)/\Lambda$, which becomes negligible when Λ is large w.r.t. ω.

Remark 5.1. *SC-SPARC constructions generally have a 'seed' to jumpstart decoding. In [8], a small fraction of sections of β are fixed a priori — this pinning condition is used to analyze the state evolution equations via the potential function method. Analogously, in the construction in [12], additional rows are introduced in the design matrix for the blocks corresponding to the first row of the base matrix. In an (ω, Λ) base matrix, the fact that the number of rows in the base matrix exceeds the number of columns by $(\omega - 1)$ helps decoding start from both ends. The asymptotic state evolution equations in Sec. 5.3.1 show how AMP decoding progresses in an (ω, Λ) base matrix.*

5.2   AMP decoder for spatially coupled SPARCs

The decoder wishes to recover the message vector $\beta \in \mathbb{R}^{ML}$ from the channel output sequence $y \in \mathbb{R}^n$, given by

$$
y = A\beta + w, \tag{5.5}
$$

where the noise vector $w \sim_{i.i.d.} \mathcal{N}N(0, \sigma^2)$.

The procedure to derive an Approximate Message Passing (AMP) decoding algorithm for SC-SPARCs is similar to that for standard SPARCs (Section 3.4, p. 39), with modifications to account

for the different variances for the blocks of A specified by the base matrix. The AMP decoder intitializes β^0 to the all-zero vector, and for $t \geq 0$, iteratively computes

$$z^t = y - A\beta^t + \widetilde{\mathsf{b}}^t \odot z^{t-1} \tag{5.6}$$

$$\beta^{t+1} = \eta^t(\beta^t + (\widetilde{S}^t \odot A)^* z^t). \tag{5.7}$$

Here $\odot$ denotes the Hadamard (entry-wise) product. The denoising function η^t, and $\widetilde{\mathsf{b}}^t \in \mathbb{R}^n$, $\widetilde{S}^t \in \mathbb{R}^{n \times ML}$ are defined below in terms of the state evolution parameters.

5.2.1   State evolution for SC-SPARCs

We recall from Section 3.2 that state evolution is a scalar recursion (see (3.25)–(3.26)) that captures the decoding progression of the AMP decoder for standard SPARCs. The key difference in SC-SPARCs is that the decoding progression depends on the row block index $r \in [L_R]$ and column block index $c \in [L_C]$. Consequently, the state evolution parameters for SC-SPARCs will also depend on the row block index $r \in [L_R]$ and the column block index $c \in [L_C]$. In detail, the state evolution (SE) iteratively computes vectors $\phi^t \in \mathbb{R}^{L_R}$ and $\psi^t \in \mathbb{R}^{L_C}$ as follows. Initialize $\psi_c^0 = 1$ for $c \in [L_C]$, and for $t = 0, 1, \ldots$, compute

$$\phi_r^t = \sigma^2 + \frac{1}{L_C} \sum_{c=1}^{L_C} W_{rc} \psi_c^t, \qquad r \in [L_R], \tag{5.8}$$

$$\psi_c^{t+1} = 1 - \mathcal{E}(\tau_c^t), \qquad c \in [L_C], \tag{5.9}$$

where

$$\tau_c^t = \frac{R}{\ln M} \left[\frac{1}{L_R} \sum_r \frac{W_{rc}}{\phi_r^t} \right]^{-1}, \tag{5.10}$$

and $\mathcal{E}(\tau_c^t)$ is defined as

$$\mathcal{E}(\tau_c^t) = \mathbb{E} \left[\frac{e^{U_1/\sqrt{\tau_c^t}}}{e^{U_1/\sqrt{\tau_c^t}} + e^{-\frac{1}{\tau_c^t}} \sum_{j=2}^{M} e^{U_j/\sqrt{\tau_c^t}}} \right], \tag{5.11}$$

with $U_1, \ldots, U_M \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$.

We define the entries of the vector $\mathsf{b}^t \in \mathbb{R}^{L_R}$ and the matrix $S^t \in \mathbb{R}^{L_R \times L_C}$ as

$$\mathsf{b}_r^t = \frac{(\phi_r^t - \sigma^2)}{\phi_r^{t-1}}, \qquad S_{rc}^t = \frac{\tau_c^t}{\phi_r^t}, \qquad \text{for } r \in [L_R], \, c \in [L_C]. \tag{5.12}$$

The vector $\widetilde{\mathsf{b}}^t \in \mathbb{R}^n$ in (5.6) is obtained by repeating M_R times each entry of b^t. Similarly, $\widetilde{S}^t \in \mathbb{R}^{n \times ML}$ in (5.7) is obtained by repeating each entry of S^t in an $M_R \times M_C$ matrix.

The denoising function $\eta^t = (\eta_1^t, \ldots, \eta_{ML}^t) : \mathbb{R}^{ML} \to \mathbb{R}^{ML}$ in (5.7) is defined as follows. For $j \in [ML]$ such that $j \in \sec(\ell)$ and the section ℓ is in the cth column block,

$$\eta_j^t(s) = \frac{e^{s_j/\tau_c^t}}{\sum_{j' \in \sec(\ell)} e^{s_{j'}/\tau_c^t}}. \tag{5.13}$$

As in the case of standard SPARCs, $\eta_j^t(s)$ depends on all the components of s in the section containing j.

5.2.2   Interpretation of the AMP decoder

The input to $\eta^t(\cdot)$ in (5.13) can be viewed as a noisy version of β. In particular, the cth block of $s^t = s$ is approximately distributed as $\beta_c + \sqrt{\tau_c^t} Z_c$, where $Z_c \in \mathbb{R}^{M_R}$ is a standard normal random vector independent of β. (Here $\beta_c \in \mathbb{R}^{M_C}$ is the part of the message vector corresponding to column block c of the design matrix.) Under the above distributional assumption, the denoising function η_j in (5.13) is the minimum mean squared error (MMSE) estimator for β_j, i.e.,

$$\eta_j^t(s) = \mathbb{E}\left[\beta_j | s = \beta_c + \sqrt{\tau_c^t}\, Z_c\right], \qquad \text{for } j \in [ML],$$

where the expectation is calculated over β and Z, with the location of the non-zero entry in each section of β being uniformly distributed within the section.

The entries of the modified residual z^t in (5.6) are approximately Gaussian and independent, with the variance determined by the block index. For $r \in [L_R]$, the SE parameter ϕ_r^t approximates the variance of the rth block of the residual $z_r^t \in \mathbb{R}^{M_R}$. The 'Onsager' term $\widetilde{\mathsf{b}}^t \odot z^{t-1}$ in (5.6) reflects the block-wise structure of z^t. Finally, the parameter ψ_c^t approximates the normalized mean-squared error in the estimate of β_c. This is discussed in the next section.

To summarize, the key difference from the state evolution parameters for standard SPARCs is that the variances of the effective observation and the residual now depend on the column- and row-block indices, respectively. These variances are captured by $\{\tau_c^t\}_{c \in [L_C]}$ and $\{\phi_r^t\}_{r \in [L_R]}$.

5.3   Measuring the performance of the AMP decoder

The performance of a SPARC decoder is measured by the *section error rate*, defined as

$$\mathcal{E}_{\text{sec}} := \frac{1}{L} \sum_{\ell=1}^{L} \mathbf{1}\{\widehat{\beta}_{\text{sec}(\ell)} \neq \beta_{\text{sec}(\ell)}\}. \tag{5.14}$$

The section error rate can be shown to be bounded by the normalized mean squared error (NMSE) as follows.

$$\mathcal{E}_{\text{sec}} \leq \frac{4}{L}\|\beta^T - \beta\|^2 = 4\left[\frac{1}{L_C} \sum_{c=1}^{L_C} \frac{\|\beta_c^T - \beta_c\|_2^2}{L/L_C}\right], \tag{5.15}$$

where in the last expression, we have written the total NMSE as an average over the NMSEs of the L_C blocks of the message vector.

Figure 5.2 shows that ψ^t closely tracks the NMSE of each block of the message vector, i.e., $\psi_c^t \approx \frac{\|\beta_c^t - \beta_c\|_2^2}{L/L_C}$ for $c \in [L_C]$. We additionally observe from the figure that as AMP iterates, the NMSE reduction propagates from the ends towards the center blocks. This decoding propagation phenomenon can be explained using an asymptotic characterization of the state evolution equations.
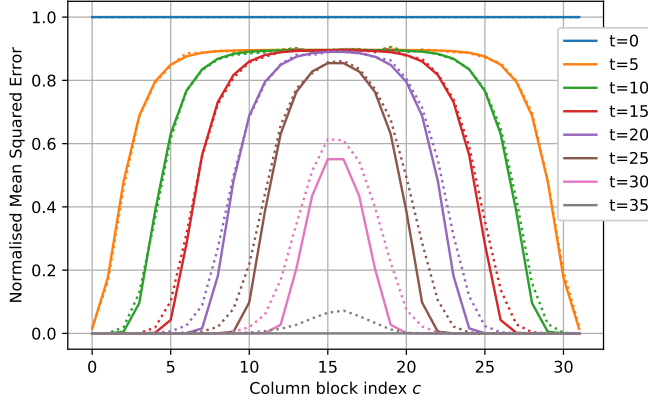
79

Figure 5.2: NMSE $\frac{\|\beta_c^t - \beta_c\|_2^2}{L/L_C}$ vs. column block index $c \in [L_C]$ for several iteration numbers. The SC-SPARC with an (ω, Λ) base matrix uses parameters: $R = 1.5$ bits, $\mathcal{C} = 2$ bits, $\omega = 6$, $\Lambda = 32$, $M = 512$, $L = 2048$ and $n = 12284$. The solid lines are the SE predictions from (5.9), and the dotted lines are the average NMSE over 100 instances of AMP decoding.

5.3.1 Asymptotic State Evolution analysis

Note that $\mathcal{E}(\tau_c^t)$ in (5.11) takes a value in $[0, 1]$. If $\mathcal{E}(\tau_c^t) = 1$, then $\psi_c^{t+1} = 0$, which means that the sections in column block c are expected to decode correctly. If we terminate the AMP decoder at iteration T, we want $\psi_c^T = 0$, for $c \in [L_C]$, so that the entire message vector is expected to decode correctly. The condition under which $\mathcal{E}(\tau_c^t)$ equals 1 in the large system limit is specified by the following lemma.

Lemma 5.2. *In the limit as the section size $M \to \infty$, the expectation $\mathcal{E}(\tau_c^t)$ in (5.11) converges to either 1 or 0 as follows.*

$$\lim_{M \to \infty} \mathcal{E}(\tau_c^t) = \begin{cases} 1 & \text{if } \frac{1}{L_R} \sum_{r=1}^{L_R} \frac{W_{rc}}{\phi_r^t} > 2R \\ 0 & \text{if } \frac{1}{L_R} \sum_{r=1}^{L_R} \frac{W_{rc}}{\phi_r^t} < 2R. \end{cases} \tag{5.16}$$

This results in the following asymptotic state evolution recursion. Initialise $\bar{\psi}_c^0 = 1$, for $c \in [L_C]$, and for $t = 0, 1, 2, \ldots$,

$$\bar{\phi}_r^t = \sigma^2 + \frac{1}{L_C} \sum_{c=1}^{L_C} W_{rc} \bar{\psi}_c^t, \qquad r \in [L_R], \tag{5.17}$$

$$\bar{\psi}_c^{t+1} = 1 - \mathbf{1}\left\{ \frac{1}{L_R} \sum_{r=1}^{L_R} \frac{W_{rc}}{\bar{\phi}_r^t} > 2R \right\}, \qquad c \in [L_C], \tag{5.18}$$

where $\bar{\phi}, \bar{\psi}$ indicate asymptotic values.

Proof. Recalling the definition of τ_c^t from (5.10), we write $\frac{1}{\tau_c^t} = \nu_c^t \ln M$, where

$$\nu_c^t = \frac{1}{RL_R} \sum_{r=1}^{L_R} \frac{W_{rc}}{\phi_r^t} \tag{5.19}$$

80

is an order 1 quantity because $\frac{1}{L_R} \sum_{r=1}^{L_R} W_{rc}$ is bounded above and below by positive constants. Therefore,

$$\mathcal{E}(\tau_c^t) = \mathbb{E}\left[\frac{e^{\sqrt{\nu_c^t \ln M} U_1}}{e^{\sqrt{\nu_c^t \ln M} U_1} + M^{-\nu_c^t} \sum_{j=2}^{M} e^{\sqrt{\nu_c^t \ln M} U_j}} \right], \tag{5.20}$$

which is in the same form as the expectation in (3.80). Therefore, following the steps in Section 3.6.1, we conclude that

$$\lim_{M \to \infty} \mathcal{E}(\tau_c^t) = \begin{cases} 1 & \text{if } \nu_c^t > 2 \\ 0 & \text{if } \nu_c^t < 2. \end{cases} \tag{5.21}$$

The proof is completed by substituting the value of ν_c^t from (5.19) in (5.21). □

Remark 5.2. *Using the definition of τ_c^t from (5.10), we can also write (5.16) as*

$$\lim_{M \to \infty} \mathcal{E}(\tau_c^t) = \begin{cases} 1 & \text{if } \tau_c^t \ln M < \frac{1}{2} \\ 0 & \text{if } \tau_c^t \ln M > \frac{1}{2}. \end{cases} \tag{5.22}$$

Remark 5.3. *Lemma 5.2 is a generalization of Lemma 3.3, the asymptotic SE result for standard SPARCs The term $\frac{1}{L_R} \sum_r \frac{W_{rc}}{\phi_r^t}$ in (5.16) represents the average signal to effective noise ratio at iteration t for the column index c. If this quantity exceeds the prescribed threshold of 2R, then the cth block of the message vector, β_c, will be decoded at the next iteration in the large system limit, i.e., $\psi_c^{t+1} = 0$.*

The asymptotic SE recursion (5.17)-(5.18) is given for a general base matrix W. We now apply it to the (ω, Λ) base matrix introduced in Definition 5.1.

Lemma 5.3. *The asymptotic SE recursion (5.17)-(5.18) for an (ω, Λ) base matrix W is as follows. Initialise $\bar{\psi}_c^0 = 1 \ \forall \ c \in [\Lambda]$, and for $t = 0, 1, 2, \ldots$,*

$$\bar{\phi}_r^t = \sigma^2 \left(1 + \frac{\kappa \cdot snr}{\omega} \sum_{c=\underline{c}_r}^{\bar{c}_r} \bar{\psi}_c^t \right), \quad r \in [\Lambda + \omega - 1], \tag{5.23}$$

$$\bar{\psi}_c^{t+1} = 1 - \mathbf{1}\left\{ \frac{P}{\omega} \sum_{r=c}^{c+\omega-1} \frac{1}{\bar{\phi}_r^t} > 2R \right\}, \quad c \in [\Lambda], \tag{5.24}$$

where $\kappa = \frac{\Lambda+\omega-1}{\Lambda}$, $snr = \frac{P}{\sigma^2}$, and

$$(\underline{c}_r, \bar{c}_r) = \begin{cases} (1, r) & \text{if } 1 \leq r \leq \omega \\ (r - \omega + 1, r) & \text{if } \omega \leq r \leq \Lambda \\ (r - \omega + 1, \Lambda) & \text{if } \Lambda \leq r \leq \Lambda + \omega - 1. \end{cases} \tag{5.25}$$

Proof. Substitute the value of W_{rc} from (5.3), and $L_C = \Lambda$, $L_R = \Lambda + \omega - 1$ in (5.17)-(5.18). □

Observe that the $\bar{\phi}_r^t$'s and $\bar{\psi}_c^t$'s are symmetric about the middle indices, i.e. $\bar{\phi}_r^t = \bar{\phi}_{L_R-r+1}^t$ for $r \leq \lfloor \frac{L_R}{2} \rfloor$ and $\bar{\psi}_c^t = \bar{\psi}_{L_C-c+1}^t$ for $c \leq \lfloor \frac{L_C}{2} \rfloor$.

81

Lemma 5.3 gives insight into the decoding progression for a large SC-SPARC defined using an (ω, Λ) base matrix. On initialization $(t = 0)$, the value of $\bar{\phi}_r^0$ for each r depends on the number of non-zero entries in row r of W, which is equal to $\bar{c}_r - \underline{c}_r + 1$, with $\bar{c}_r, \underline{c}_r$ given by (5.25). Therefore, $\bar{\phi}_r^0$ increases from $r = 1$ until $r = \omega$, is constant for $\omega \le r \le \Lambda$, and then starts decreasing again after $r = \Lambda$. As a result, $\bar{\psi}_c^1$ is smallest for c at either end of the base matrix ($c \in \{1, \Lambda\}$) and increases as c moves towards the middle, since the $\sum_{r=c}^{c+\omega-1}(\bar{\phi}_r^0)^{-1}$ term in (5.24) is largest for $c \in \{1, \Lambda\}$, followed by $c \in \{2, \Lambda - 1\}$, and so on. Therefore, we expect the blocks of the message vector corresponding to column index $c \in \{1, \Lambda\}$ to be decoded most easily, followed by $c \in \{2, \Lambda - 1\}$, and so on. Fig. 5.2 shows that this is indeed the case.

The decoding progression for subsequent iterations shown in Fig. 5.2 can be explained using Lemma 5.3 by tracking the evolution of the $\bar{\phi}_r^t$'s and $\bar{\psi}_c^t$'s. In particular, one finds that if column c^* decodes in iteration t, i.e. $\bar{\psi}_{c^*}^t = 0$, then columns within a coupling width away (i.e., columns $c \in \{c^* - (\omega - 1), \ldots, c^* + (\omega - 1)\}$) will become easier to decode in iteration $(t + 1)$.

In the following, with a slight abuse of terminology, we will use the phrase "column c is decoded in iteration t" to mean $\bar{\psi}_c^t = 0$.

Proposition 5.4. *[59] Consider an SC-SPARC constructed using an (ω, Λ) base matrix with rate $R < \frac{1}{2\kappa} \ln(1 + \kappa \cdot snr)$, where $\kappa = \frac{\Lambda + \omega - 1}{\Lambda}$. (Note that $\frac{1}{2\kappa} \ln(1 + \kappa \cdot snr) \in [\mathcal{C}/\kappa, \mathcal{C}].$) Then, according to the asymptotic state evolution equations in Lemma 5.3, the following statements hold in the large system limit:*

1. *The AMP decoder will be able to start decoding if*

$$\omega > \left(\frac{1}{e^{2R\kappa} - 1} - \frac{1}{\kappa \cdot snr} \right)^{-1}. \tag{5.26}$$

2. *If (5.26) is satisfied, then the sections in the first and last c^* blocks of the message vector will be decoded in the first iteration (i.e. $\bar{\psi}_c^1 = 0$ for $c \in \{1, 2, \ldots, c^*\} \cup \{\Lambda - c^* + 1, \Lambda - c^* + 2, \ldots, \Lambda\}$), where c^* is bounded from below as*

$$c^* \ge \min\left\{ (\omega - 1), \left\lfloor \omega \cdot \frac{1 + \kappa \cdot snr}{(\kappa \cdot snr)^2} \cdot [\ln(1 + \kappa \cdot snr) - 2R\kappa] \right\rfloor \right\}. \tag{5.27}$$

3. *At least $2c^*$ additional columns will decode in each subsequent iteration until the message is fully decoded. Therefore, the AMP decoder will fully decode in at most $\left\lceil \frac{\Lambda}{2c^*} \right\rceil$ iterations.*

Remark 5.4. *The proposition implies that for any rate $R < \mathcal{C}$, AMP decoding is successful in the large system limit, i.e., $\bar{\psi}_c^T = 0$ for all $c \in [\Lambda]$. Indeed, consider a rate $R = \mathcal{C}/\kappa_0$, for any constant $\kappa_0 > 1$. Then choose ω to satisfy (5.26) (with κ replaced by κ_0), and Λ large enough that $\kappa = \frac{\Lambda + \omega - 1}{\Lambda} \le \kappa_0$. With this choice of (ω, Λ) and rate R, the conditions of the proposition are satisfied, and hence, all the columns decode in the large system limit.*

Remark 5.5. *The proof of the proposition shows that if $R < \frac{snr}{2(1 + \kappa \cdot snr)}$, then $\bar{\psi}_c^1 = 0$, for all $c \in [\Lambda]$, i.e., the entire codeword decodes in the first iteration.*

82

Remark 5.6. *The state evolution recursion was analyzed for a certain class of spatially coupled SPARCs by Barbier et al. [8] using the potential method introduced in [120, 76, 35]. It is shown in [8] that the fixed points of the state evolution recursion (5.8)–(5.9) coincide with the stationary points of a suitably defined potential function. This is then used to show 'threshold saturation' for spatially coupled SPARCs with AMP decoding, i.e., for all rates $R < C$, state evolution predicts vanishing probability of decoding error in the limit of large section size. In contrast, Proposition 5.4 establishes threshold saturation by directly characterizing the decoding progression in the large system limit.*

Remark 5.7. *A non-asymptotic version of Proposition 5.4, which describes the decoding progression for large but finite M, can be found in [96, Sec. IV].*

Remark 5.8. *For a fixed rate $R < C$, one can establish a bound similar to Theorem 3.3 on the probability of excess section error rate of an AMP decoded spatially coupled SPARC. This requires two technical ingredients in addition to Proposition 5.4. The first is a conditional distribution lemma similar to Lemma 3.8, but tailored to the spatially coupled design matrix. In particular, the conditional distributions of the vectors h^{t+1} and b^t now depend on the column block and row block indices, respectively. These conditional distributions are then used to establish a concentration result similar to Lemma 3.10 which shows that the NMSE in each iteration $\frac{1}{L}\|\beta - \beta^t\|^2$ is tracked with high probability by the state evolution quantity $\frac{1}{L_C}\sum_c \psi_c^t$. Proposition 5.4 guarantees that this quantity is small after $\left\lceil \frac{\Lambda}{2c^*} \right\rceil$ iterations. The rigorous performance analysis of AMP for spatially coupled SPARCs using the above ingredients will be detailed in a forthcoming paper.*

Proof of Proposition 5.4. Since the $\bar{\phi}_r^t$'s and $\bar{\psi}_c^t$'s in (5.23) and (5.24) are symmetric about the middle indices, we will only consider decoding the first half of the columns, $c \in \{1, \ldots, \lfloor \frac{\Lambda+1}{2} \rfloor\}$, and the same arguments will apply to the second half by symmetry.

For column c to decode in iteration 1, i.e., for $\bar{\psi}_c^1 = 0$, we require the argument of the indicator function in (5.24) to be satisfied for $t = 0$, which corresponds to

$$F_c := \frac{\kappa \cdot \text{snr}}{\omega} \sum_{r=c}^{c+\omega-1} \frac{1}{1 + \frac{\kappa \cdot \text{snr}}{\omega} \cdot (\overline{c}_r - \underline{c}_r + 1)} > 2R\kappa. \tag{5.28}$$

1) Since the F_c is largest for column $c = 1$, (5.28) must be satisfied with $c = 1$ for *any* column to start decoding. Moreover, using (5.25), we find

$$F_1 = \frac{\kappa \cdot \text{snr}}{\omega} \sum_{r=1}^{\omega} \frac{1}{1 + \frac{\kappa \cdot \text{snr}}{\omega} \cdot r} \overset{(i)}{>} \int_{\frac{\kappa \cdot \text{snr}}{\omega}}^{\frac{\kappa \cdot \text{snr}}{\omega}(\omega+1)} \frac{1}{1 + x} \, dx$$

$$= \ln\left(1 + \frac{\kappa \cdot \text{snr}}{1 + \kappa \cdot \text{snr} \cdot \frac{1}{\omega}}\right), \tag{5.29}$$

where the inequality (i) is obtained by using left Riemann sums on the decreasing function $\frac{1}{1+x}$. Using (5.29) in (5.28), we conclude that if $\ln\left(1 + \frac{\kappa \cdot \text{snr}}{1+\kappa \cdot \text{snr}/\omega}\right) > 2R\kappa$, then column $c = 1$ will decode in the first iteration. Rearranging this inequality yields (5.26).

2) Given an (ω, Λ) pair that satisfies (5.26), we can find a lower bound on the total number of columns that decode in the first iteration. In order to decode column c (and column $\Lambda - c + 1$ by symmetry) in the first iteration, we require (5.28) to be satisfied. For $c < \omega$, this condition corresponds to

$$F_c = \frac{\kappa \cdot \text{snr}}{\omega} \left[\left(\sum_{r=c}^{\omega-1} \frac{1}{1 + \frac{\kappa \cdot \text{snr}}{\omega} \cdot r} \right) + \frac{c}{1 + \kappa \cdot \text{snr}} \right] > 2R\kappa, \tag{5.30}$$

and for columns $c \in \{\omega, \ldots, \Lambda - \omega + 1\}$, the condition in (5.28) becomes

$$\frac{\text{snr}}{1 + \kappa \cdot \text{snr}} > 2R, \tag{5.31}$$

where (5.25) was used to find the values of $\underline{c}_r$ and $\overline{c}_r$. Since F_c defined in (5.28) is smallest for columns $c \in \{\omega, \ldots, \Lambda - \omega + 1\}$, all columns decode in the first iteration if (5.31) is satisfied.

For columns $c < \omega$, we can obtain a lower bound for F_c:

$$F_c = \frac{\kappa \cdot \text{snr}}{\omega} \left[\left(\sum_{r=c}^{\omega-1} \frac{1}{1 + \frac{\kappa \cdot \text{snr}}{\omega} \cdot r} \right) + \frac{c}{1 + \kappa \cdot \text{snr}} \right]$$

$$\overset{(i)}{>} \int_{\frac{\kappa \cdot \text{snr}}{\omega} c}^{\frac{\kappa \cdot \text{snr}}{\omega} \omega} \frac{1}{1 + x} \, dx + \frac{c}{\omega} \frac{\kappa \cdot \text{snr}}{(1 + \kappa \cdot \text{snr})}$$

$$= \ln\left(1 + \kappa \cdot \text{snr}\right) - \ln\left(1 + \kappa \cdot \text{snr} \cdot \frac{c}{\omega}\right) + \frac{c}{\omega} \frac{\kappa \cdot \text{snr}}{(1 + \kappa \cdot \text{snr})}$$

$$\overset{(ii)}{>} \ln\left(1 + \kappa \cdot \text{snr}\right) - \kappa \cdot \text{snr} \cdot \frac{c}{\omega} + \frac{c}{\omega} \frac{\kappa \cdot \text{snr}}{(1 + \kappa \cdot \text{snr})}$$

$$= \ln\left(1 + \kappa \cdot \text{snr}\right) - \frac{c}{\omega} \frac{(\kappa \cdot \text{snr})^2}{(1 + \kappa \cdot \text{snr})}, \tag{5.32}$$

where (i) is obtained by using left Riemann sums on the decreasing function $\frac{1}{1+x}$, and (ii) from $\ln(x) \leq x - 1$. Therefore, if the RHS of (5.32) is greater than $2R\kappa$ then (5.30) is satisfied, and column c will decode in the first iteration. This inequality corresponds to

$$c < \omega \cdot \frac{1 + \kappa \cdot \text{snr}}{(\kappa \cdot \text{snr})^2} \cdot \left[\ln\left(1 + \kappa \cdot \text{snr}\right) - 2R\kappa\right]. \tag{5.33}$$

In other words, all columns $c < \omega$ that also satisfy (5.33) will decode in the first iteration. Therefore, the number of columns (in the first half) that decode in the first iteration, denoted c^*, can be bounded from below by (5.27).

3) We want to prove that if the first (and last) c^* columns decode in the first iteration, then at least the first (and last) tc^* columns will decode by iteration t, for $t \geq 1$. We look at the $c^* < \omega$ case because all columns would have been decoded in the first iteration if $c^* \geq \omega$. We again only consider the first half of the columns (and rows) due to symmetry.

We prove by induction. The $t = 1$ case holds by the previous statement that the first c^* columns decode in the first iteration. From (5.30), this corresponds to the following inequality being satisfied:

$$\frac{\text{snr}}{\omega} \left[\left(\sum_{r=c^*}^{\omega-1} \frac{1}{1 + \frac{\kappa \cdot \text{snr}}{\omega} \cdot r} \right) + \frac{c^*}{1 + \kappa \cdot \text{snr}} \right] > 2R. \tag{5.34}$$

84

Assume that the statement holds for some $t \geq 1$, i.e. $\bar{\psi}_c^t = 0$ for $c \in [tc^*]$. We assume that $tc^* < \lfloor \frac{\Lambda+1}{2} \rfloor$, otherwise all the columns will have already been decoded. Then, from (5.23), we obtain

$$
\bar{\phi}_r^t \leq \begin{cases} \sigma^2, & 1 \leq r \leq tc^*, \\ \sigma^2 \left(1 + \frac{\kappa \cdot \text{snr}}{\omega}(r - tc^*)\right), & tc^* < r < tc^* + \omega, \\ \sigma^2 \left(1 + \kappa \cdot \text{snr}\right), & tc^* + \omega \leq r \leq \lfloor \frac{\Lambda+\omega-1}{2} \rfloor + 1. \end{cases} \tag{5.35}
$$

(We have a $\leq$ sign in (5.35) rather than an equality because indices r near $\frac{\Lambda+\omega-1}{2}$ may have smaller values in the final iterations, due to columns from the other half and within ω indices away having already been decoded.)

We now show that the statement holds for $(t+1)$, i.e., $\psi_c^{t+1} = 0$ for columns $c \in [(t+1)c^*]$. In order for columns $c \in \{tc^* + 1, \ldots, (t+1)c^*\}$ to decode in iteration $(t+1)$, the inequality in the indicator function in (5.24) must be satisfied when $c = (t+1)c^*$ (the LHS of the inequality is larger for $c \in \{tc^* + 1, \ldots, (t+1)c^* - 1\}$). This corresponds to

$$
\frac{\text{snr}}{\omega} \left[\left(\sum_{r=(t+1)c^*}^{tc^*+\omega-1} \frac{1}{1 + \frac{\kappa \cdot \text{snr}}{\omega}(r - tc^*)} \right) + \frac{c^*}{1 + \kappa \cdot \text{snr}} \right] > 2R, \tag{5.36}
$$

which is equivalent to (5.34), noting that $(t+1)c^* < tc^* + \omega$ since $c^* < \omega$. Therefore, (5.36) holds by the condition, and the statement holds for $(t+1)$. Due to symmetry, the same arguments can be applied to the last tc^* and $(t+1)c^*$ columns. Therefore, at least c^* columns from each half will decode in every iteration. □

5.4   Simulation results

We evaluate the empirical performance of SC-SPARCs constructed from (ω, Λ) base matrices. As in Chapter 4, we use a Hadamard based design matrix (instead of a Gaussian one), which gives significant reductions in running time and required memory, with very similar error performance to Gaussian design matrices.

Figure 5.3 compares the average section error rate (SER) and the codeword error rate of spatially coupled SPARCs with standard (non-SC) SPARCs, both with and without power allocation (PA). The code length is the same for all the codes, and the power allocation was designed using the iterative algorithm described in Section 4.2.1. AMP decoding is used for all the codes. Comparing standard SPARCs without PA and SC-SPARCs, we see that spatial coupling significantly improves the error performance: the rate threshold below which the SER drops steeply to a negligible value is higher for SC-SPARCs. We also observe that at rates close to the channel capacity, standard SPARCs with PA have lower SER than SC-SPARCs. However, as the rate decreases, the drop in SER for standard SPARCs with PA is not as steep as that for SC-SPARCs.

With respect to codeword error rate, we observe that SC-SPARCs significantly outperform non-SC SPARCs with power allocation. This is because power allocated SPARCs tend to have a much
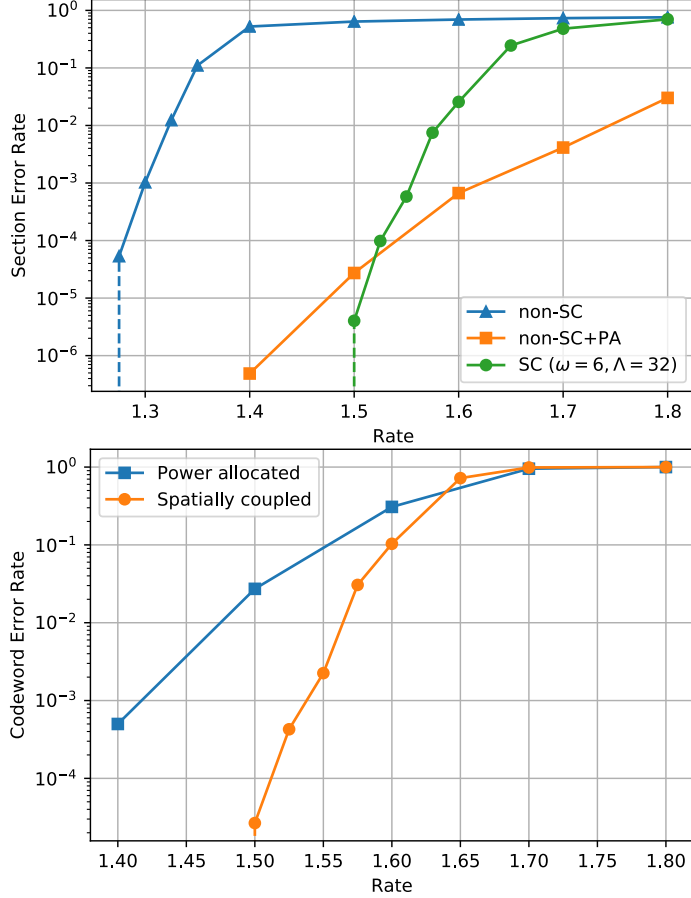
Figure 5.3: Average section error rate (top) and codeword error rate (bottom) vs. rate at snr = 15, $\mathcal{C} = 2$ bits. The SPARC parameters are $M = 512$, $L = 1024$, $n \in [5100, 7700]$. The graph at the top shows plots for non-SC SPARCs with and without power allocation, and SC-SPARCs with an (ω, Λ) base matrix with $\omega = 6, \Lambda = 32$. The code length is the same for the three cases. The dotted vertical lines indicate that no section errors were observed over at least 10^4 trials at smaller rates.

larger number of trials with at least one section error; the number of section errors in such trials is typically small, the errors occur mostly in the sections with low power. In contrast, it was observed that the SC-SPARC had many fewer trials with codeword errors, but when a codeword error occurred, it often resulted a large number of sections were in error.

Next, we examine the effect of changing the coupling width ω. Fig. 5.4 compares the average SER of SC-SPARCs with (ω, Λ) base matrices with $\Lambda = 32$ and varying ω. For a fixed Λ, we observe from (5.4) that a larger ω requires a larger inner SPARC rate R_{inner} for the same overall SC-SPARC rate R. A larger value of R_{inner} makes decoding harder; on the other hand increasing the coupling width ω helps decoding. Thus for a given rate R, there is a trade-off: as illustrated by Fig. 5.4, increasing ω improves the SER up to a point, but the performance degrades for larger ω. In general, ω should be large enough so that coupling can benefit decoding, but not so large that R_{inner} is very close to the channel capacity. For example, for $R = 1.6$ bits and $\Lambda = 32$, the inner SPARC rate $R_{\text{inner}} = 1.65, 1.75, 1.85, 1.95$ bits for $\omega = 2, 4, 6, 8$, respectively. With the capacity being $\mathcal{C} = 2$

86

Figure 5.4: Average section error rate vs. rate at snr $= 15$, $\mathcal{C} = 2$ bits, $M = 512$, $L = 1024$, $n \in [5100, 6200]$. Plots are shown for SC-SPARCs with an (ω, Λ) base matrix with $\Lambda = 32$ and $\omega \in \{2, 4, 6, 8\}$. For a given rate, the code length is the same for different ω values. The dotted vertical line indicates that for $\omega = 6$ and 8, no section errors were observed over 10^4 trials at $R = 1.5$ bits.

bits, the figure shows that $\omega = 6$ is the best choice for $R = 1.6$ bits, with $\omega = 8$ being noticeably worse. This also indicates that smaller values ω would be favored as the rate R gets closer to $\mathcal{C}$.

Part II

Lossy Compression with SPARCs

Chapter 6

Optimal Encoding

In the second part of this monograph, we turn our focus to SPARCs for lossy compression. Developing practical codes for lossy compression at rates approaching Shannon's rate-distortion bound has been a long-standing goal in information theory. A practical compression code requires a codebook with low storage complexity as well as encoding and decoding algorithms with low computational complexity. The storage complexity of a SPARC is proportional to the size of the size of the design matrix, which is polynomial is the code length n.

In this chapter, we analyze the compression performance of SPARCs with optimal encoding. The performance is measured via the squared error distortion criterion. Though the complexity of the optimal encoder grows exponentially in the code length, its performance sets a benchmark for efficient SPARC encoders (like the one discussed in the next chapter).

SPARCs were first considered for lossy compression by Kontoyiannis et al. in [68], where some some preliminary results on their compression performance were presented. Here we will discuss the analysis in [112] and [115] which shows that for i.i.d. Gaussian sources, SPARCs with minimum-distance encoding attain the optimal rate-distortion function and the optimal excess-distortion exponent.

6.1   Problem set-up

The source sequence is denoted by $s = (s_1, \ldots, s_n)$, and the reconstruction sequence by $\hat{s} = (s_1, \ldots, s_n)$. The distortion is measured by the normalized squared error $\frac{1}{n}\|s - \hat{s}\|^2$. Throughout this chapter, for any vector $x \in \mathbb{R}^n$, we will use the notation $|x|$ to denote the normalized norm $\|x\|/\sqrt{n}$.

Codebook construction   The sparse regression codebook is as described in Section 1.1. Each codeword is of the form $A\beta$, where the design matrix A has entries $\sim_{i.i.d.} \mathcal{N}(0, \frac{1}{n})$. The codeword is determined by the vector $\beta \in \mathcal{B}_{M,L}$, which has one non-zero in each section.

The main difference from the channel coding construction is that the values of the non-zeros in β do not have to satisfy a power constraint — they can be chosen in any way to help the compression encoder. In this chapter, we set all the non-zero values to be equal:

$$c_1 = \ldots = c_L = \sqrt{\frac{nc^2}{L}}, \tag{6.1}$$

where the value of c is specified later in (6.16)

As there are M^L codewords, to obtain a compression rate of R nats/sample we need

$$M^L = e^{nR}. \tag{6.2}$$

In this chapter, we choose $M = L^b$ for some $b > 1$ so that (6.2) implies

$$L \log L = \frac{nR}{b}. \tag{6.3}$$

Thus L is $\Theta\left(n/\log n\right)$, and the number of columns ML in the dictionary A is $\Theta\left((n/\log n)^{b+1}\right)$, a polynomial in n.

Minimum-distance encoder   The optimal encoder for squared-error distortion is the minimum-distance encoder. For the SPARC, it is defined by a mapping $g : \mathbb{R}^n \to \mathcal{B}_{M,L}$, which produces the β that produces the codeword closest to the source sequence in Euclidean distance, i.e.,

$$\hat{\beta} = g(s) = \underset{\beta \in \mathcal{B}_{M,L}}{\operatorname{argmin}} \|s - A\beta\|.$$

Decoder   This is a mapping $h : \mathcal{B}_{M,L} \to \mathbb{R}^n$. On receiving $\hat{\beta} \in \mathcal{B}_{M,L}$ from the encoder, the decoder produces the reconstruction $h(\hat{\beta}) = A\hat{\beta}$.

Performance measures   For a rate-distortion code $\mathcal{C}_n$ with code length n and encoder and decoder mappings g, h, the probability of excess distortion at distortion level D is

$$P_e(\mathcal{C}_n, D) = P\left(|s - h(g(s))|^2 > D\right). \tag{6.4}$$

For a SPARC, the probability measure in (6.4) is with respect to the random source sequence s and the random design matrix A.

Definition 6.1. *A rate R is achievable at distortion level D if there exists a sequence of rate R codes $\{\mathcal{C}_n\}_{n=1,2,\ldots}$ such that $\lim_{n\to\infty} P_e(\mathcal{C}_n, D) = 0$. The infimum of all rates achievable at distortion level D by any sequence of codes is the Shannon rate-distortion function, denoted by $R^*(D)$.*

A rate R is achievable by SPARCs if there exists a sequence of rate R SPARCs $\{\mathcal{C}_n\}_{n=1,2,\ldots}$, with $\mathcal{C}_n$ defined by an $n \times L_n M_n$ design matrix whose parameter L_n satisfies (6.3) with a fixed b and $M_n = L_n^b$.

For an i.i.d. Gaussian source where $s_1, s_2, \ldots$ are $\sim_{i.i.d.} \mathcal{N}(0, \sigma^2)$, the Shannon rate-distortion function is [32]

$$R^*(D) = \begin{cases} \frac{1}{2} \log \frac{\sigma^2}{D} & D < \sigma^2, \\ 0 & D \geq \sigma^2. \end{cases} \qquad (6.5)$$

The excess-distortion exponent at distortion-level D of a sequence of rate R codes $\{\mathcal{C}_n\}_{n=1,2,\ldots}$ is given by

$$r(D, R) = -\limsup_{n \to \infty} \frac{1}{n} \log P_e(\mathcal{C}_n, D), \qquad (6.6)$$

where $P_e(\mathcal{C}_n, D)$ is defined in (6.4). The optimal excess-distortion exponent for a rate-distortion pair (R, D) is the supremum of the excess-distortion exponents over all sequences of codes with rate R at distortion-level D.

The optimal excess-distortion exponent for discrete memoryless sources was obtained by Marton [81], and for memoryless Gaussian sources by Ihara and Kubo [60].

Theorem 6.1. *[60] For an i.i.d. Gaussian source distributed as $\mathcal{N}(0, \sigma^2)$ and squared-error distortion criterion, the optimal excess-distortion exponent at rate R and distortion-level D is*

$$r^*(D, R) = \begin{cases} \frac{1}{2} \left(\frac{a^2}{\sigma^2} - 1 - \log \frac{a^2}{\sigma^2} \right) & R > R^*(D) \\ 0 & R \leq R^*(D) \end{cases} \qquad (6.7)$$

where $a^2 = De^{2R}$.

For $R > R^*(D)$, the exponent in (6.7) is the Kullback-Leibler divergence between two zero-mean Gaussians, distributed as $\mathcal{N}(0, a^2)$ and $\mathcal{N}(0, \sigma^2)$, respectively.

6.2   Performance of the optimal decoder

The key result in this chapter (Theorem 6.2) is a large deviations bound on the excess distortion probability of a SPARC. This result is then used to show that SPARCs attain the optimal rate-distortion function and excess-distortion exponent for i.i.d. Gaussian sources.

For $x > 1$, let

$$b_{\min}(x) = \frac{28R\, x^4}{\left(1 + \frac{1}{x}\right)^2 \left(1 - \frac{1}{x}\right) \left[-1 + \left(1 + \frac{2\sqrt{x}}{(x-1)} \left(R - \frac{1}{2}(1 - \frac{1}{x})\right)\right)^{1/2} \right]^2} \qquad (6.8)$$

Theorem 6.2. *[115] Let the source sequence $s = (s_1, \ldots, s_n)$ be drawn from an ergodic source with mean zero and variance σ^2. Let $D \in (0, \sigma^2)$, $R > \frac{1}{2} \log \frac{\sigma^2}{D}$, and $\gamma^2 \in (\sigma^2, De^{2R})$. Let*

$$b > \max\left\{ 2,\ b_{min}\left(\gamma^2/D\right) \right\}, \qquad (6.9)$$

where $b_{min}(.)$ is defined in (6.8). Let $\mathcal{C}_n$ be SPARC of rate R defined via an $n \times L_n M_n$ design matrix with $M_n = L_n^b$ and L_n determined by (6.3). Then the probability of excess distortion for $\mathcal{C}_n$ at distortion level D satisfies

$$P_e(\mathcal{C}_n, D) \le P\left(\frac{\|s\|^2}{n} \ge \gamma^2\right) + \exp\left(-\kappa n^{1+c}\right), \tag{6.10}$$

where κ, c are strictly positive universal constants.

The proof of the theorem is given in Section 6.3.

The first term on the RHS of (6.10) is the probability that the empirical second moment of the source exceeds γ^2. This probability does not depend on the codebook. The second term is a bound on the conditional probability of not finding a SPARC codeword within distortion D given that $\frac{\|s\|^2}{n} < \gamma^2$. Since the second term decays faster than exponentially in n, for large n the excess distortion probability in (6.10) is dominated by the first term.

Let us compare the bound in (6.10) with the excess distortion probability of a Shannon-style random i.i.d. codebook with optimal encoding. The first term remains unchanged as it does not depend on the codebook. The second term, which is the probability of not finding a codeword within distortion D for a source sequence with $\frac{\|s\|^2}{n} < \gamma^2$, decays *double exponentially* in n [60] for the random i.i.d. codebook. Though the second term decays much faster for an i.i.d. codebook than for SPARCs, for large n the excess distortion probability is still dominated by the first term. We therefore expect the excess-distortion exponent of a SPARC to be the same as that of a random i.i.d. codebook. We also know that a sequence of random i.i.d. codebooks attains the optimal exponent in (6.7); hence, based on the previous claim a sequence of SPARCs would also attain the optimal exponent. This is made precise in the following corollary.

Corollary 6.2. *Let s be drawn from an i.i.d. Gaussian source with mean zero and variance σ^2. Fix rate $R > \frac{1}{2}\log\frac{\sigma^2}{D}$, and let $a^2 = De^{2R}$. Fix any $\epsilon \in (0, a^2 - \sigma^2)$, and*

$$b > \max\left\{2, b_{min}\left(\frac{a^2 - \epsilon}{D}\right)\right\}. \tag{6.11}$$

There exists a sequence of rate R SPARCs with parameter b that achieves the excess-distortion exponent

$$\frac{1}{2}\left(\frac{a^2 - \epsilon}{\sigma^2} - 1 - \log\frac{a^2 - \epsilon}{\sigma^2}\right).$$

Consequently:

1. *SPARCs attain the Shannon rate-distortion function of an i.i.d. Gaussian source.*

2. *The supremum of excess-distortion exponents achievable by SPARCs for i.i.d. Gaussian sources sources is equal to the optimal one, given by (6.7).*

Proof. From Theorem 6.2, we know that for any $\epsilon \in (0, a^2 - \sigma^2)$, there exists a sequence of rate R SPARCs $\{C_n\}$ for which

$$P_e(C_n, D) \leq P(|s|^2 \geq a^2 - \epsilon) \left(1 + \frac{\exp(-\kappa n^{1+c})}{P(|s|^2 \geq a^2 - \epsilon)}\right) \tag{6.12}$$

for sufficiently large n, as long as the parameter b satisfies (6.11). For s that is i.i.d. $\mathcal{N}(0, \sigma^2)$, Cramér's large deviation theorem [34] yields

$$\lim_{n \to \infty} -\frac{1}{n} \log P(|s|^2 \geq a^2 - \epsilon) = \frac{1}{2} \left(\frac{a^2 - \epsilon}{\sigma^2} - 1 - \log \frac{a^2 - \epsilon}{\sigma^2}\right) \tag{6.13}$$

for $(a^2 - \epsilon) > \sigma^2$. Thus $P(|s|^2 \geq a^2 - \epsilon)$ decays exponentially with n; in comparison $\exp(-\kappa n^{1+c})$ decays *faster* than exponentially with n. Therefore, from (6.12), the excess-distortion exponent satisfies

$$
\begin{aligned}
&\liminf_{n \to \infty} \frac{-1}{n} \log P_e(C_n, D) \\
&\geq \liminf_{n \to \infty} \frac{-1}{n} \left[\log P(|s|^2 \geq a^2 - \epsilon) + \log\left(1 + \frac{\exp(-\kappa n^{1+c})}{P(|s|^2 \geq a^2 - \epsilon)}\right)\right] \\
&= \frac{1}{2} \left(\frac{a^2 - \epsilon}{\sigma^2} - 1 - \log \frac{a^2 - \epsilon}{\sigma^2}\right).
\end{aligned}
\tag{6.14}
$$

Since $\epsilon > 0$ can be chosen arbitrarily small, the supremum of all achievable excess-distortion exponents is $\frac{1}{2}\left(\frac{a^2}{\sigma^2} - 1 - \log \frac{a^2}{\sigma^2}\right)$, which is optimal from Fact 6.1. $\qquad \square$

Theorem 6.2 and Corollary 6.2 together show that sparse regression codes are essentially as good as random i.i.d Gaussian codebooks in terms of rate-distortion function, excess-distortion exponent, and robustness. By robustness, we mean that a SPARC designed to compress an i.i.d Gaussian source with variance σ^2 to distortion D can compress any ergodic source with variance at most σ^2 to distortion D. This property is also satisfied by random i.i.d Gaussian codebooks [77, 99, 100]. Moreover, Lapidoth [77] also showed that for any ergodic source, with an i.i.d. Gaussian random codebook one cannot attain a mean-squared distortion smaller than the distortion-rate function of an i.i.d Gaussian source with the same variance.

To sum up, the sparse regression ensemble has good covering properties, with the advantage of much smaller codebook storage complexity than the i.i.d random ensemble (polynomial vs. exponential in block-length).

The remainder of this chapter is devoted to proving Theorem 6.2. The proof involves using the second moment method and Suen's inequality [63] to show that if $|s|^2 \leq \gamma^2$, then with high probability there exists at least one codeword within distortion D of the source sequence. Proving the result turns out to be significantly easier in the regime where $R > R_0(D)$ where

$$R_0(D) := \max\left\{\frac{1}{2} \log \frac{\sigma^2}{D}, \left(1 - \frac{D}{\sigma^2}\right)\right\}. \tag{6.15}$$

The rate $R_0(D)$ in (6.15) is equal to $R^*(D)$ when $\frac{D}{\sigma^2} \leq x^*$, but is strictly larger than $R^*(D)$ when $\frac{D}{\sigma^2} > x^*$, where $x^* \approx 0.203$; see Fig. 6.1.
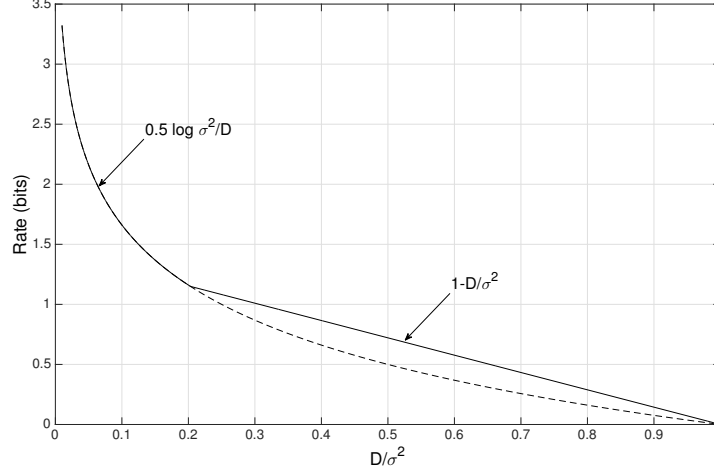
Figure 6.1: The solid line shows the previous achievable rate $R_0(D)$, given in (6.15). The rate-distortion function $R^*(D)$ is shown in dashed lines. It coincides with $R_0(D)$ for $D/\sigma^2 \leq x^*$, where $x^* \approx 0.203$.

The reason for the result being harder to prove for $R \in (R^*(D), R_0(D)]$ is discussed on p.101 after introducing the key elements of the proof. Roughly speaking, at these low rates the probability of the SPARC codebook having an atypically large number of codewords within distortion D of the source sequence is high, and so a standard application of second moment method fails.

6.3 Proof of Theorem 6.2

Fix a rate $R > R^*(D)$, and b greater than the minimum value specified by the theorem. Note that $De^{2R} > \sigma^2$ since $R > \frac{1}{2} \log \frac{\sigma^2}{D}$. Let γ^2 be any number such that $\sigma^2 < \gamma^2 < De^{2R}$.

Code Construction: Fix block length n and section parameter b. Then pick L as specified by (6.3) and $M = L^b$. Construct an $n \times ML$ design matrix A with entries drawn i.i.d. $\mathcal{N}(0, 1/n)$. The codebook consists of all vectors $A\beta$ such that $\beta \in \mathcal{B}_{M,L}$. The non-zero entries of β are all set equal to a value specified below.

Encoding and Decoding: If the source sequence s is such that $|s|^2 \geq \gamma^2$, then the encoder declares an error. If $|s|^2 \leq D$, then s can be trivially compressed to within distortion D using the all-zero codeword. The addition of this extra codeword to the codebook affects the rate in a negligible way.

If $|s|^2 \in (D, \gamma^2)$, then s is compressed in two steps. First, quantize $|s|^2$ with an n-level uniform scalar quantizer $Q(.)$ with support in the interval $(D, \gamma^2]$. Conveying the scalar quantization index to the decoder (with an additional $\log n$ nats) allows us to adjust the codebook variance according to the norm of the observed source sequence.[1] The non-zero entries of β are each set to $\sqrt{nc^2/L}$,

---

[1]The scalar quantization step is only included to simplify the analysis. In fact, we could use the same codebook variance $(\gamma^2 - D)$ for all s that satisfy $|s|^2 \leq (\gamma^2 - D)$, but this would make the forthcoming large deviations analysis quite cumbersome.

where

$$c^2 = Q(|s|^2) - D. \tag{6.16}$$

so that each SPARC codeword has variance $c^2 = (Q(|s|^2) - D)$. Define a 'quantized-norm' version of s as

$$\tilde{s} := \sqrt{\frac{Q(|s|^2)}{|s|^2}}\, s. \tag{6.17}$$

Note that $|\tilde{s}|^2 = Q(|s|^2)$. We use the SPARC to compress $\tilde{s}$. The encoder finds

$$\hat{\beta} := \operatorname*{argmin}_{\beta \in \mathcal{B}_{M,L}} \|\tilde{s} - A\beta\|^2.$$

The decoder receives $\hat{\beta}$ and reconstructs $\hat{s} = A\hat{\beta}$. Note that for block length n, the total number of bits transmitted by encoder is $\log n + L \log M$, yielding an overall rate of $R + \frac{\log n}{n}$ nats/sample.

Let $\mathcal{E}(\tilde{s})$ be the event that the minimum of $|\tilde{s} - A\beta|^2$ over $\beta \in \mathcal{B}_{M,L}$ is greater than D. The encoder declares an error if $\mathcal{E}(\tilde{s})$ occurs. If $\mathcal{E}(\tilde{s})$ *does not* occur, it can be verified that the overall distortion can be bounded as

$$\left|s - A\hat{\beta}\right|^2 \le D + \frac{\kappa}{n}, \tag{6.18}$$

for some positive constant κ. The overall rate (including that of the scalar quantizer) is $R + \frac{\log n}{n}$.

Denoting the probability of excess distortion for this code by $P_{e,n}$, we have

$$P_{e,n} \le P(|s|^2 \ge \gamma^2) + \max_{\rho^2 \in (D, \gamma^2)} P(\mathcal{E}(\tilde{s}) \mid |\tilde{s}|^2 = \rho^2). \tag{6.19}$$

To bound the second term in (6.19), without loss of generality we can assume that the source sequence is $\tilde{s} = (\rho, \dots, \rho)$. This is because the codebook distribution is rotationally invariant, due to the i.i.d. $\mathcal{N}(0,1)$ design matrix A. For any β, the entries of $A\beta(i)$ i.i.d. $\mathcal{N}(0, \rho^2 - D)$. We enumerate the codewords as $A\beta(i)$, where $\beta(i) \in \mathcal{B}_{M,L}$ for $i = 1, \dots, e^{nR}$.

Define the indicator random variables

$$U_i(\tilde{s}) = \begin{cases} 1 & \text{if } |A\beta(i) - \tilde{s}|^2 \le D, \\ 0 & \text{otherwise.} \end{cases} \tag{6.20}$$

We can then write

$$P(\mathcal{E}(\tilde{s})) = P\left(\sum_{i=1}^{e^{nR}} U_i(\tilde{s}) = 0\right). \tag{6.21}$$

For a fixed $\tilde{s}$, the $U_i(\tilde{s})$'s are dependent. Indeed, if $\beta(i)$ and $\beta(j)$ overlap in r of their non-zero positions, then the column sums forming codewords $\hat{s}(i)$ and $\hat{s}(j)$ will share r common terms, and consequently $U_i(\tilde{s})$ and $U_j(\tilde{s})$ will be dependent.

For brevity, we henceforth denote $U_i(\tilde{s})$ by just U_i. We also write $X := \sum_{i=1}^{e^{nR}} U_i$. We refer to β_i as a *solution* if $U_i = 1$. Hence X is the number of solutions.

97

To highlight the main ideas in the proof, before obtaining a non-asymptotic bound for the probability in (6.21), we will first prove the following asymptotic result.

$$P(X > 0) = P\left(\sum_{i=1}^{e^{nR}} U_i > 0\right) \to 1 \text{ as } n \to \infty. \tag{6.22}$$

We will first apply the second moment method (second MoM) to prove (6.22), and then use Suen's correlation inequality to prove the non-asymptotic result in the statement of the theorem.

For any non-negative random variable X, the second MoM bounds the probability of the event $X > 0$ from below as

$$P(X > 0) \geq \frac{(\mathbb{E}X)^2}{\mathbb{E}[X^2]}. \tag{6.23}$$

The inequality (6.23) follows from the Cauchy-Schwarz inequality

$$(\mathbb{E}[XY])^2 \leq \mathbb{E}X^2 \, \mathbb{E}Y^2$$

by substituting $Y = \mathbf{1}_{\{X>0\}}$. To apply it to our setting, we first observe that

$$\mathbb{E}[X^2] = \mathbb{E}\left[X\sum_{i=1}^{e^{nR}} U_i\right] = \sum_{i=1}^{e^{nR}} \mathbb{E}[XU_i] = \sum_{i=1}^{e^{nR}} P(U_i = 1)\mathbb{E}[X|U_i = 1]$$

$$= \mathbb{E}X \cdot \mathbb{E}[X \,|\, U_1 = 1]. \tag{6.24}$$

Using (6.24) in (6.23), we obtain

$$P(X > 0) \geq \frac{\mathbb{E}X}{\mathbb{E}[X \,|\, U_1 = 1]}. \tag{6.25}$$

6.3.1 Second moment method computations

To compute $\mathbb{E}X$, we derive a general lemma specifying the probability that a randomly chosen i.i.d $\mathcal{N}(0, y)$ codeword is within distortion z of a source sequence s with $|s|^2 = x$. This lemma will be used in other parts of the proof as well.

Lemma 6.3. *Let s be a vector with $|s|^2 = x$. Let $\hat{s}$ be an i.i.d. $\mathcal{N}(0, y)$ random vector that is independent of s. Then for $x, y, z > 0$ and sufficiently large n, we have*

$$\frac{\kappa}{\sqrt{n}} e^{-nf(x,y,z)} \leq P\left(|\hat{s} - s|^2 \leq z\right) \leq e^{-nf(x,y,z)}, \tag{6.26}$$

where κ is a universal positive constant and for $x, y, z > 0$, the large-deviation rate function f is

$$f(x, y, z) = \begin{cases} \frac{x+z}{2y} - \frac{xz}{Ay} - \frac{A}{4y} - \frac{1}{2}\ln\frac{A}{2x} & \text{if } z \leq x + y, \\ 0 & \text{otherwise,} \end{cases} \tag{6.27}$$

and

$$A = \sqrt{y^2 + 4xz} - y. \tag{6.28}$$

Proof. We have

$$P\left(|\hat{s} - s|^2 \leq z\right) = P\left(\frac{1}{n}\sum_{k=1}^{n}(\hat{s}_k - s_k)^2 \leq z\right) = P\left(\frac{1}{n}\sum_{k=1}^{n}(\hat{s}_k - \sqrt{x})^2 \leq z\right), \tag{6.29}$$

where the last equality is due to the rotational invariance of the distribution of $\hat{s}$, i.e., $\hat{s}$ has the same joint distribution as $O\hat{s}$ for any orthogonal (rotation) matrix O. In particular, we choose O to be the matrix that rotates s to the vector $(\sqrt{x}, \ldots, \sqrt{x})$, and note that $|\hat{s} - s|^2 = |O\hat{s} - Os|^2$. Then, using the strong version of Cramér's large deviation theorem due to Bahadur and Rao [34, 7], we have

$$\frac{\kappa}{\sqrt{n}}e^{-nI(x,y,z)} \leq P\left(\frac{1}{n}\sum_{k=1}^{n}(\hat{s}_k - x)^2 \leq z\right) \leq e^{-nI(x,y,z)}, \tag{6.30}$$

where the large-deviation rate function I is given by

$$I(x,y,z) = \sup_{\lambda \geq 0}\left\{\lambda z - \log\mathbb{E}e^{\lambda(\hat{S} - \sqrt{x})^2}\right\}. \tag{6.31}$$

The expectation on the RHS of (6.31) is computed with $\hat{S} \sim \mathcal{N}(0, y)$. Using standard calculations, we obtain

$$\log\mathbb{E}e^{\lambda(\hat{S} - \sqrt{x})^2} = \frac{\lambda x}{1 - 2y\lambda} - \frac{1}{2}\log(1 - 2y\lambda), \qquad \lambda < 2y. \tag{6.32}$$

Substituting the expression in (6.32) in (6.31) and maximizing over $\lambda \in [0, 2y)$ yields $I(x,y,z) = f(x,y,z)$, where f is given by (6.27). $\qquad\square$

The expected number of solutions is given by

$$\mathbb{E}X = e^{nR}P(U_1 = 1) = e^{nR}P\left(|A\beta(1) - \tilde{s}|^2 \leq D\right). \tag{6.33}$$

Since $\tilde{s} = (\rho, \rho, \ldots, \rho)$, and $A\beta(1)$ is i.i.d. $\mathcal{N}(0, \rho^2 - D)$, applying Lemma 6.3 we obtain the bounds

$$\frac{\kappa}{\sqrt{n}}e^{nR}e^{-nf(\rho^2, \rho^2 - D, D)} \leq \mathbb{E}X \leq e^{nR}e^{-nf(\rho^2, \rho^2 - D, D)}, \tag{6.34}$$

Note that

$$f(\rho^2, \rho^2 - D, D) = \frac{1}{2}\log\frac{\rho^2}{D}. \tag{6.35}$$

Next consider $\mathbb{E}[X \,|\, U_1 = 1]$. If $\beta(i)$ and $\beta(j)$ overlap in r of their non-zero positions, the column sums forming codewords $\hat{s}(i)$ and $\hat{s}(j)$ will share r common terms. Therefore,

$$\mathbb{E}[X \,|\, U_1 = 1] = \sum_{i=1}^{e^{nR}}P(U_i = 1 \,|\, U_1 = 1) = \sum_{i=1}^{e^{nR}}\frac{P(U_i = 1, U_1 = 1)}{P(U_1 = 1)}$$

$$\stackrel{(a)}{=} \sum_{r=0}^{L}\binom{L}{r}(M-1)^{L-r}\frac{P(U_2 = U_1 = 1 \,|\, \mathcal{F}_{12}(r))}{P(U_1 = 1)} \tag{6.36}$$

99

where $\mathcal{F}_{12}(r)$ is the event that the codewords corresponding to U_1 and U_2 share r common terms. In (6.36), (a) holds because for each codeword $\hat{s}(i)$, there are a total of $\binom{L}{r}(M-1)^{L-r}$ codewords which share exactly r common terms with $\hat{s}(i)$, for $0 \leq r \leq L$.

From (6.36) and (6.33), the key ratio in (6.24) is

$$
\begin{aligned}
\frac{\mathbb{E}[X \mid U_1 = 1]}{\mathbb{E}X} &= \sum_{r=0}^{L} \binom{L}{r}(M-1)^{L-r} \frac{P(U_2 = U_1 = 1 \mid \mathcal{F}_{12}(r))}{e^{nR} \left(P(U_1 = 1)\right)^2} \\
&\stackrel{(a)}{\sim} 1 + \sum_{\alpha = \frac{1}{L}, \dots, \frac{L}{L}} \binom{L}{L\alpha} \frac{P(U_2 = U_1 = 1 \mid \mathcal{F}_{12}(\alpha))}{M^{L\alpha} \left(P(U_1 = 1)\right)^2} \\
&\stackrel{(b)}{=} 1 + \sum_{\alpha = \frac{1}{L}, \dots, \frac{L}{L}} e^{n\Delta_\alpha}
\end{aligned}
\tag{6.37}
$$

where (a) is obtained by substituting $\alpha = \frac{r}{L}$ and $e^{nR} = M^L$. The notation $x_L \sim y_L$ means that $x_L / y_L \to 1$ as $L \to \infty$. The equality (b) is from [112, Appendix A], where it is shown that

$$
\Delta_\alpha \leq \frac{\kappa}{L} + \frac{R}{b} \min\{\alpha, \bar{\alpha}, \tfrac{\log 2}{\log L}\} - h(\alpha)
\tag{6.38}
$$

where $\kappa > 0$ is a universal constant, and

$$
h(\alpha) := \alpha R - \frac{1}{2} \log \left(\frac{1+\alpha}{1 - \alpha(1 - \frac{2D}{\rho^2})} \right).
\tag{6.39}
$$

The term $e^{n\Delta_\alpha}$ in (6.37) may be interpreted as follows. Conditioned on $U_1 = 1$, i.e. $\beta(1)$ is a solution, the expected number of solutions that share αL common terms with $\beta(1)$ is $\sim e^{n\Delta_\alpha}\mathbb{E}X$. Recall that we require the left side of (6.37) to tend to 1 as $n \to \infty$. Therefore, we need $\Delta_\alpha < 0$ for $\alpha = \frac{1}{L}, \dots, \frac{L}{L}$. From (6.38), we need $h(\alpha)$ to be positive in order to guarantee that $\Delta_\alpha < 0$.

It can be shown [112, Appendix A] that for $R > (1 - D/\rho^2)$, the function $h(\alpha) = \alpha R - g(\rho^2)$ is strictly positive in the interval $[\frac{1}{L}, \frac{L-1}{L}]$. Further, for all sufficiently large L its minimum in the interval is attained at $\alpha = 1/L$ where it equals

$$
h(1/L) = \frac{1}{L} \left(R - (1 - D/\rho^2) \right) + \frac{\kappa}{L^2}, \quad \kappa > 0.
\tag{6.40}
$$

Using this bound for $h(\alpha)$ in (6.38), we find that if $R > (1 - D/\rho^2)$, and

$$
b > \frac{2.5R + \frac{\kappa}{\log L}}{R - (1 - D/\rho^2) + \frac{\kappa}{L}},
\tag{6.41}
$$

then the exponent Δ_α in (6.37) is strictly negative for $\frac{1}{L} \leq \alpha \leq \frac{(L-1)}{L}$. Consequently, the key ratio $\frac{\mathbb{E}[X \mid U_1 = 1]}{\mathbb{E}X} \to 1$ as $n \to \infty$.

However, when $\frac{1}{2} \log \frac{\rho^2}{D} < R < (1 - \frac{D}{\rho^2})$, it can be verified that $h(\alpha) < 0$ for $\alpha \in (0, \alpha^*)$ where $\alpha^* \in (0, 1)$ is the solution to $h(\alpha) = 0$. Thus Δ_α is *positive* for $\alpha \in (0, \alpha^*)$ when $\frac{1}{2} \log \frac{\rho^2}{D} < R \leq (1 - \frac{D}{\rho^2})$.

100

Consequently, (6.37) implies that

$$\frac{\mathbb{E}[X \mid U_1 = 1]}{\mathbb{E}X} \sim \sum_\alpha e^{n\Delta_\alpha} \to \infty \quad \text{as} \quad n \to \infty, \tag{6.42}$$

so the second MoM fails.

The reason for the failure of the second MoM in the regime $R < (1 - \frac{D}{\rho^2})$ is due to the size-biasing induced by conditioning on $U_1 = 1$. Indeed, for any s, there are atypical realizations of the design matrix that yield a very large number of solutions. The total probability of these matrices is small enough that $\mathbb{E}X$ in not significantly affected by these realizations. However, conditioning on β being a solution increases the probability that the realized design matrix is one that yields an unusually large number of solutions. At low rates, the conditional probability of the design matrix being atypical is large enough to make $\mathbb{E}[X|U_1 = 1] \gg \mathbb{E}X$, causing the second MoM to fail.

The key to rectifying the second MoM failure is to show that $X(\beta) \approx \mathbb{E}X$ with high probability *although* $\mathbb{E}[X|U_1 = 1] \gg \mathbb{E}X$. We then apply the second MoM to count just the 'good' solutions, i.e., solutions $\beta(i)$ for which $X|_{U_i=1} \approx \mathbb{E}X$. This approach was first used in the work of Coja-Oghlan and Zdeborová [28] on finding sharp thresholds for two-coloring of random hypergraphs.

6.3.2 Refining the second moment method

Given that $\beta \in \mathcal{B}_{M,L}$ is a solution, for $\alpha = 0, \frac{1}{L}, \ldots, \frac{L}{L}$ define $X_\alpha(\beta)$ to be the number of solutions that share αL non-zero terms with β. The *total* number of solutions given that β is a solution is

$$X(\beta) = \sum_{\alpha=0,\frac{1}{L},\ldots,\frac{L}{L}} X_\alpha(\beta) \tag{6.43}$$

Using this notation, we have

$$\frac{\mathbb{E}[X \mid U_1 = 1]}{\mathbb{E}X} \overset{(a)}{=} \frac{\mathbb{E}[X(\beta)]}{\mathbb{E}X}$$
$$= \sum_{\alpha=0,\frac{1}{L},\ldots,\frac{L}{L}} \frac{\mathbb{E}[X_\alpha(\beta)]}{\mathbb{E}X} \overset{(b)}{\sim} 1 + \sum_{\alpha=\frac{1}{L},\ldots,\frac{L}{L}} e^{n\Delta_\alpha}, \tag{6.44}$$

where (a) holds because the symmetry of the code construction allows us to condition on a generic $\beta \in \mathcal{B}_{M,L}$ being a solution; (b) follows from (6.37).

The key ingredient in the proof is the following lemma, which shows that $X_\alpha(\beta)$ is much smaller than $\mathbb{E}X$ w.h.p $\forall \alpha \in \{\frac{1}{L}, \ldots, \frac{L}{L}\}$. In particular, $X_\alpha(\beta) \ll \mathbb{E}X$ *even* for α for which

$$\frac{\mathbb{E}[X_\alpha(\beta)]}{\mathbb{E}X} \sim e^{n\Delta_\alpha} \to \infty \quad \text{as } n \to \infty.$$

Lemma 6.4. *[115, Lemma 4] Let $R > \frac{1}{2}\log\frac{\rho^2}{D}$. If $\beta \in \mathcal{B}_{M,L}$ is a solution, then for sufficiently large L*

$$P\left(X_\alpha(\beta) \le L^{-5/2}\,\mathbb{E}X, \quad \text{for } \frac{1}{L} \le \alpha \le \frac{L-1}{L}\right) \ge 1 - \eta \tag{6.45}$$

101

where

$$\eta = L^{-3.5\left(\frac{b}{b_{min}(\rho^2/D)}-1\right)}.$$

(6.46)

The function $b_{min}(.)$ is defined in (6.8).

We refer the reader to [115] for the proof. The probability measure in Lemma 6.4 is the conditional distribution on the space of design matrices A given that β is a solution.

Definition 6.5. *For $\epsilon > 0$, call a solution β "ϵ-good" if*

$$\sum_{\alpha=\frac{1}{L},\dots,\frac{L}{L}} X_\alpha(\beta) < \epsilon\,\mathbb{E}X.$$

(6.47)

Since we have fixed $\tilde{s} = (\rho,\dots,\rho)$, whether a solution β is ϵ-good or not is determined by the design matrix. Lemma 6.4 guarantees that w.h.p. any solution β will be ϵ-good, i.e., if β is a solution, w.h.p. the design matrix is such that the number of solutions sharing any common terms with β is less $\epsilon\mathbb{E}[X]$.

The key to proving the asymptotic result in (6.22) is to apply the second MoM only to ϵ-good solutions. Fix $\epsilon = L^{-1.5}$. For $i = 1,\dots, e^{nR}$, define the indicator random variables

$$V_i = \begin{cases} 1 & \text{if } |A\beta(i) - \tilde{s}|^2 \le D \text{ and } \beta(i) \text{ is } \epsilon\text{-good,} \\ 0 & \text{otherwise.} \end{cases}$$

(6.48)

The number of ϵ-good solutions, denoted by X_g, is given by

$$X_g = V_1 + V_2 + \dots + V_{e^{nR}}.$$

(6.49)

We will apply the second MoM to X_g to show that $P(X_g > 0) \to 1$ as $n \to \infty$. We have

$$P(X_g > 0) \ge \frac{(\mathbb{E}X_g)^2}{\mathbb{E}[X_g^2]} = \frac{\mathbb{E}X_g}{\mathbb{E}[X_g\,|\,V_1 = 1]}$$

(6.50)

where the second equality is obtained by writing $\mathbb{E}[X_g^2] = (\mathbb{E}X_g)\mathbb{E}[X_g\,|\,V_1 = 1]$, similar to (6.24).

Lemma 6.6. *a) $\mathbb{E}X_g \ge (1-\eta)\mathbb{E}X$, where η is defined in (6.46).*

b) $\mathbb{E}[X_g\,|\,V_1 = 1] \le (1 + L^{-0.5})\mathbb{E}X$.

Proof. Due to the symmetry of the code construction, we have

$$\mathbb{E}X_g = e^{nR}P(V_1 = 1) \overset{(a)}{=} e^{nR}P(U_1 = 1)P(V_1 = 1|U_1 = 1)$$
$$= \mathbb{E}X \cdot P(\beta(1) \text{ is } \epsilon\text{-good } |\,\beta(1) \text{ is a solution}).$$

(6.51)

In (6.51), (a) follows from the definitions of V_i in (6.48) and U_i in (6.20). Given that $\beta(1)$ is a solution, Lemma 6.4 shows that

$$\sum_{\alpha=\frac{1}{L},\dots,\frac{L}{L}} X_\alpha(\beta(1)) < (\mathbb{E}X)L^{-1.5}.$$

(6.52)

102

with probability at least $1 - \eta$. As $\epsilon = L^{-1.5}$, $\beta(1)$ is ϵ-good according to Definition 6.5 if (6.52) is satisfied. Thus $\mathbb{E}X_g$ in (6.51) can be lower bounded as

$$\mathbb{E}X_g \geq (1 - \eta)\mathbb{E}X. \tag{6.53}$$

For part (b), first observe that the total number of solutions X is an upper bound for the number of ϵ-good solutions X_g. Therefore

$$\mathbb{E}[X_g | V_1 = 1] \leq \mathbb{E}[X | V_1 = 1]. \tag{6.54}$$

Given that $\beta(1)$ is an ϵ-good solution, the expected number of solutions can be expressed as

$$\begin{aligned}
&\mathbb{E}[X | V_1 = 1] \\
&= \mathbb{E}[X_0(\beta(1)) | V_1 = 1] + \mathbb{E}[\sum_{\alpha = \frac{1}{L}, \ldots, \frac{L}{L}} X_\alpha(\beta(1)) | V_1 = 1].
\end{aligned} \tag{6.55}$$

There are $(M - 1)^L$ codewords that share no common terms with $\beta(1)$, and are thus independent of the event $V_1 = 1$.

$$\begin{aligned}
\mathbb{E}[X_0(\beta(1)) | V_1 = 1] &= \mathbb{E}[X_0(\beta(1))] = (M - 1)^L P(|\tilde{s} - A\beta|^2 \leq D) \\
&\leq M^L P(|\tilde{s} - A\beta|^2 \leq D) = \mathbb{E}X.
\end{aligned} \tag{6.56}$$

Next, note that conditioned on $\beta(1)$ being an ϵ-good solution (i.e., $V_1 = 1$),

$$\sum_{\alpha = \frac{1}{L}, \ldots, \frac{L}{L}} X_\alpha(\beta(1)) < \epsilon\,\mathbb{E}X \tag{6.57}$$

with certainty. This follows from the definition of ϵ-good in (6.47). Using (6.56) and (6.57) in (6.55), we conclude that

$$\mathbb{E}[X | V_1 = 1] < (1 + \epsilon)\mathbb{E}X. \tag{6.58}$$

Combining (6.58) with (6.54) completes the proof of Lemma 6.6. $\qquad\square$

Using Lemma 6.6 in (6.50), we obtain

$$P(X_g > 0) \geq \frac{\mathbb{E}X_g}{\mathbb{E}[X_g | V_1 = 1]} \geq \frac{(1 - \eta)}{1 + \epsilon} = \frac{1 - L^{-3.5(\frac{b}{b_{min}(\rho^2/D)} - 1)}}{1 + L^{-3/2}}, \tag{6.59}$$

where the last equality is obtained by using the definition of η in (6.46) and $\epsilon = L^{-0.5}$. Hence the probability of the existence of at least one good solution tends to 1 as $L \to \infty$. Therefore $P(X > 0)$ in (6.22) also tends to one.

6.3.3 A non-asymptotic bound for $P(X = 0)$

We now prove the result (6.10) by obtaining a non-asymptotic bound for $P(X_g = 0)$. In contrast to (6.59) which proves that $P(X_g = 0)$ decays polynomially in L, we will use Suen's inequality to show that this probability decays *super-exponentially* in L.

We begin with some definitions.

103

Definition 6.7 (Dependency Graphs [63])**.** *Let $\{V_i\}_{i\in\mathcal{I}}$ be a family of random variables (defined on a common probability space). A dependency graph for $\{V_i\}$ is any graph Γ with vertex set $V(\Gamma) = \mathcal{I}$ whose set of edges satisfies the following property: if A and B are two disjoint subsets of $\mathcal{I}$ such that there are no edges with one vertex in A and the other in B, then the families $\{V_i\}_{i\in A}$ and $\{V_i\}_{i\in B}$ are independent.*

Remark 6.1. *[63, Example 1.5, p.11] Suppose $\{Y_\alpha\}_{\alpha\in\mathcal{A}}$ is a family of independent random variables, and each $V_i, i \in \mathcal{I}$ is a function of the variables $\{Y_\alpha\}_{\alpha\in A_i}$ for some subset $A_i \subseteq \mathcal{A}$. Then the graph with vertex set $\mathcal{I}$ and edge set $\{ij : A_i \cap A_j \neq \emptyset\}$ is a dependency graph for $\{U_i\}_{i\in\mathcal{I}}$.*

In our setting, we fix $\epsilon = L^{-3/2}$, let V_i be the indicator the random variable defined in (6.48). Note that V_i is one if and only if $\beta(i)$ is an ϵ-good solution. The set of codewords that share at least one common term with $\beta(i)$ are the ones that play a role in determining whether $\beta(i)$ is an ϵ-good solution or not. Hence, the graph Γ with vertex set $V(\Gamma) = \{1, \ldots, e^{nR}\}$ and edge set $e(\Gamma)$ given by

$$\{ij : i \neq j \text{ and the codewords } \beta(i), \beta(j)$$
$$\text{share at least one common term}\}$$

is a dependency graph for the family $\{V_i\}_{i=1}^{e^{nR}}$.

For a given codeword $\beta(i)$, there are $\binom{L}{r}(M-1)^{L-r}$ other codewords that have exactly r terms in common with $\beta(i)$, for $0 \leq r \leq (L-1)$. Therefore each vertex in the dependency graph for the family $\{V_i\}_{i=1}^{e^{nR}}$ is connected to

$$\sum_{r=1}^{L-1} \binom{L}{r}(M-1)^{L-r} = M^L - 1 - (M-1)^L$$

other vertices.

Proposition 6.8 (Suen's Inequality [63])**.** *Let $V_i \sim Bern(p_i), i \in \mathcal{I}$, be a finite family of Bernoulli random variables having a dependency graph Γ. Write $i \sim j$ if ij is an edge in Γ. Define*

$$\lambda = \sum_{i\in\mathcal{I}} \mathbb{E}V_i, \quad \Delta = \frac{1}{2}\sum_{i\in\mathcal{I}}\sum_{j\sim i} \mathbb{E}(V_iV_j), \quad \delta = \max_{i\in\mathcal{I}} \sum_{k\sim i} \mathbb{E}V_k.$$

Then

$$P\left(\sum_{i\in\mathcal{I}} V_i = 0\right) \leq \exp\left(-\min\left\{\frac{\lambda}{2}, \frac{\lambda}{6\delta}, \frac{\lambda^2}{8\Delta}\right\}\right). \tag{6.60}$$

We apply Suen's inequality with the dependency graph specified above for $\{V_i\}_{i=1}^{e^{nR}}$ to compute an upper bound for $P(X_g = 0)$, where $X_g = \sum_{i=1}^{e^{nR}} V_i$ is the total number of ϵ-good solutions for $\epsilon = L^{-3/2}$.

First Term $\frac{\lambda}{2}$: We have

$$\lambda = \sum_{i=1}^{e^{nR}} \mathbb{E}V_i = \mathbb{E}X_g \geq \mathbb{E}X\,(1-\eta). \tag{6.61}$$

104

where the last inequality follows from Lemma 6.4, with η defined in (6.46). Using the expression from (6.33) for the expected number of solutions $\mathbb{E}X$, we have

$$\lambda \geq (1 - \eta)\frac{\kappa}{\sqrt{n}}e^{n(R - \frac{1}{2}\log\frac{\rho^2}{D})}, \tag{6.62}$$

where $\kappa > 0$ is a universal constant. For $b > b_{min}(\rho^2/D)$, (6.46) implies that η approaches 1 with growing L.

Second term $\lambda/(6\delta)$: Due to the symmetry of the code construction, we have

$$\delta = \max_{i\in\{1,\ldots,e^{nR}\}} \sum_{k\sim i} P\left(V_k = 1\right) = \sum_{k\sim i} P\left(V_k = 1\right) \quad \forall i \in \{1,\ldots,e^{nR}\}$$
$$= \left(M^L - 1 - (M-1)^L\right) P\left(V_1 = 1\right). \tag{6.63}$$

Combining this together with the fact that $\lambda = M^L P(V_1 = 1)$, we obtain

$$\frac{\lambda}{\delta} = \frac{M^L}{M^L - 1 - (M-1)^L} = \frac{1}{1 - L^{-bL} - (1 - L^{-b})^L}, \tag{6.64}$$

where the second equality is obtained by substituting $M = L^b$. Using a Taylor series bound for the denominator of (6.64) (see [112, Sec. V] for details) yields the following lower bound for sufficiently large L:

$$\frac{\lambda}{\delta} \geq \frac{L^{b-1}}{2}. \tag{6.65}$$

Third Term $\lambda^2/(8\Delta)$: We have

$$\Delta = \frac{1}{2}\sum_{i=1}^{M^L}\sum_{j\sim i}\mathbb{E}\left[V_iV_j\right] = \frac{1}{2}\sum_{i=1}^{M^L}P(V_i = 1)\sum_{j\sim i}P(V_j = 1 \mid V_i = 1)$$
$$\stackrel{(a)}{=} \frac{1}{2}\mathbb{E}X_g\sum_{j\sim 1}P(V_j = 1 \mid V_1 = 1)$$
$$= \frac{1}{2}\mathbb{E}X_g\,\mathbb{E}\left[\sum_{j\sim 1}\mathbf{1}\{V_j = 1\} \mid V_1 = 1\right] \tag{6.66}$$
$$\stackrel{(b)}{\leq} \frac{1}{2}\mathbb{E}X_g\,\mathbb{E}\left[\sum_{\alpha=\frac{1}{L},\ldots,\frac{L-1}{L}}X_\alpha(\beta(1)) \mid V_1 = 1\right].$$

In (6.66), (a) holds because of the symmetry of the code construction. The inequality (b) is obtained as follows. The number of ϵ-good solutions that share common terms with $\beta(1)$ is bounded above by the total number of solutions sharing common terms with $\beta(1)$. The latter quantity can be expressed as the sum of the number of solutions sharing exactly αL common terms with $\beta(1)$, for $\alpha \in \{\frac{1}{L},\ldots,\frac{L-1}{L}\}$.

105

Conditioned on $V_1 = 1$, i.e., the event that $\beta(1)$ is a ϵ-good solution, the total number of solutions that share common terms with $\beta(1)$ is bounded by $\epsilon \, \mathbb{E}X$. Therefore, from (6.66) we have

$$
\begin{aligned}
\Delta &\le \frac{1}{2}\mathbb{E}X_g \, \mathbb{E}\left[\sum_{\alpha = \frac{1}{L}, \ldots, \frac{L-1}{L}} X_\alpha(\beta(1)) \mid V_1 = 1 \right] \\
&\le \frac{1}{2}\left(\mathbb{E}X_g\right)\left(L^{-3/2}\,\mathbb{E}X\right) \le \frac{L^{-3/2}}{2}(\mathbb{E}X)^2,
\end{aligned}
\tag{6.67}
$$

where we have used $\epsilon = L^{-3/2}$, and the fact that $X_g \le X$. Combining (6.67) and (6.61), we obtain

$$
\frac{\lambda^2}{8\Delta} \ge \frac{(1-\eta)^2(\mathbb{E}X)^2}{4L^{-3/2}(\mathbb{E}X)^2} \ge \kappa L^{3/2},
\tag{6.68}
$$

where κ is a strictly positive constant.

Applying Suen's inequality: Using the lower bounds obtained in (6.62), (6.65), and (6.68) in (6.60), we obtain

$$
P\left(\sum_{i=1}^{e^{nR}} V_i \right) \le \exp\left(-\kappa \, \min\left\{ e^{n\left(R - \frac{1}{2}\log \frac{\rho^2}{D} - \frac{\log n}{2n}\right)}, L^{b-1}, L^{3/2} \right\} \right),
\tag{6.69}
$$

where κ is a positive constant. Recalling from (6.3) that $L = \Theta(\frac{n}{\log n})$ and $R > \frac{1}{2}\ln\frac{\rho^2}{D}$, we see that for $b > 2$,

$$
P\left(\sum_{i=1}^{e^{nR}} V_i \right) \le \exp\left(-\kappa n^{1+c} \right),
\tag{6.70}
$$

Note that the condition $b > b_{min}(\rho^2/D)$ is also required for η in Lemma 6.4 to go to 0 with growing L.

Using (6.70) in (6.19), we conclude that for any $\gamma^2 \in (\sigma^2, D e^{2R})$ the probability of excess distortion can be bounded as

$$
\begin{aligned}
P_{e,n} &\le P(|s|^2 \ge \gamma^2) + \max_{\rho^2 \in (D,\gamma^2)} P(\mathcal{E}(\tilde{s}) \mid |\tilde{s}|^2 = \rho^2) \\
&\le P(|s|^2 \ge \gamma^2) + \exp(-\kappa n^{1+c}),
\end{aligned}
\tag{6.71}
$$

provided the parameter b satisfies

$$
b > \max_{\rho^2 \in (D,\gamma^2)} \max\left\{ 2, b_{min}\left(\rho^2/D\right) \right\}.
\tag{6.72}
$$

It can be verified from the definition in (6.8) that $b_{min}(x)$ is strictly increasing in $x \in (1, e^{2R})$. Therefore, the maximum on the RHS of (6.72) is bounded by $\max\left\{ 2, b_{min}\left(\gamma^2/D\right) \right\}$. Choosing b to be larger than this value will guarantee that (6.71) holds. This completes the proof of the theorem.

Chapter 7

Computationally Efficient Encoding

In this chapter, we discuss an efficient SPARC encoder for lossy compression with squared-error distortion. The encoding algorithm is based on successive cancellation: in each iteration, one column from a section of A is chosen to be part of the codeword. The column is chosen based on a test statistic that measures the correlation of each column in the section with a residual vector.

For any ergodic source with variance σ^2, it is shown that the encoding algorithm attains the optimal Gaussian distortion-rate function $D^*(R) = \sigma^2 e^{-2R}$, for any rate $R > 0$. Furthermore, for any fixed distortion level above $D^*(R)$, the probability of excess distortion decays exponentially in the block length n. We note that for finite alphabet memoryless sources, several coding techniques have been proposed to approach the rate-distortion bound with computationally feasible encoding and decoding [66, 54, 67, 62, 53, 116, 69, 4].

We first give a heuristic derivation of the encoding algorithm and then state the main result (Theorem 7.1), a large deviations bound on the excess distortion probability. We also present numerical results to illustrate the empirical compression performance of the algorithm.

Notation: As in the previous chapter, for any vector x we write $|x|$ to denote $\|x\|/\sqrt{n}$. We also write $\langle x, y \rangle$ for the Euclidean inner product between vectors $x, y \in \mathbb{R}^n$.

7.1   Computationally efficient encoding algorithm

Consider a source sequence $s \in \mathbb{R}^n$ generated by an ergodic source with zero mean and variance σ^2. The SPARC is defined via an $n \times ML$ design matrix A with entries drawn i.i.d. $\mathcal{N}(0, 1/n)$. The codebook consists of all vectors $A\beta$ such that $\beta \in \mathcal{B}_{M,L}$. The non-entry of β in section ℓ is set to

$$c_\ell = \sqrt{2(\ln M)\sigma^2 \left(1 - \frac{2R}{L}\right)^{\ell-1}}, \quad \ell \in [L]. \tag{7.1}$$

The encoding algorithm is intialized with $r_0 = s$, and consists of L steps, defined as follows.

Step ℓ, $\ell = 1, \ldots, L$: Pick

$$m_\ell = \operatorname*{argmax}_{j:\ (\ell-1)M < j \leq \ell M} \left\langle \sqrt{n} A_j, \frac{r_{\ell-1}}{\|r_{\ell-1}\|} \right\rangle. \tag{7.2}$$

Set

$$r_\ell = r_{\ell-1} - c_\ell A_{m_\ell}, \tag{7.3}$$

where c_ℓ is given by (7.1).

The codeword $\hat{\beta}$ has non-zero values in positions m_ℓ, $1 \leq \ell \leq L$. The value of the non-zero in section ℓ given by c_ℓ.

The algorithm chooses the non-zero locations $\{m_\ell\}$ in a greedy manner (section by section) to minimize the norm of the residual r_ℓ. In the next section, we give a heuristic derivation of the algorithm, which also explains the choice of coefficients in (7.1).

Computational complexity   There are L stages in the algorithm, where each stage involves computing M inner products followed by finding the maximum among them. The complexity therefore scales as $O(nML)$; the number of operations per source sample is $O(ML)$. If we choose $M = L^b$ for some $b > 0$, then $L = \Theta\left(\frac{n}{\log n}\right)$, and the per-sample complexity is $O\left(n/\log n\right)^{b+1}$.

When we have several source sequences to be encoded in succession, the encoder can have a pipelined architecture with L modules. The first module computes the inner product of the source sequence with each column in the first section of A and determines the maximum; the second module computes the inner product of the first-step residual with each column in the second section of A, and so on. Each module has M parallel units, with each unit consisting of a multiplier and an accumulator to compute an inner product in a pipelined fashion. After an initial delay of L source sequences, all the modules work simultaneously. This encoder architecture requires computational space (memory) of the order nLM and has constant computation time per source symbol.

7.2   Heuristic derivation of the algorithm

We now present a non-rigorous analysis of the encoding algorithm based on the following observations.

1. For $1 \leq j \leq ML$, by standard concentration of measure arguments, $|A_j|^2$ is close to 1 for large n.

2. Similarly, for an ergodic source $|s|^2$ is close to σ^2 for large n.

3. For random variables $X_1, X_2 \ldots, X_M \sim_{i.i.d.} \mathcal{N}(0,1)$, the maximum $\max\{X_1, \ldots, X_M\}$ concentrates on $\sqrt{2 \ln M}$ for large M [33].

The deviations of these quantities from their typical values above are precisely characterized in the proof of the main result (Section 7.5).

We begin with the following lemma about projections of standard normal vectors.

Lemma 7.1. *Let $A_1, \ldots, A_N \in \mathbb{R}^n$ be N mutually independent random vectors with i.i.d. $\mathcal{N}(0, 1/n)$ entries. Then, for any unit norm random vector $r \in \mathbb{R}^n$ which is independent of the collection $\{A_j\}_{j=1}^N$, the inner products*

$$T_j := \left\langle \sqrt{n} A_j, r \right\rangle, \quad j = 1, \ldots, N$$

are i.i.d. $\mathcal{N}(0, 1)$ random variables that are independent of r.

The lemma is a straightforward consequence of the rotational invariance of the distribution of a standard normal vector. A proof can be found in [113, Appendix I].

Step 1: Consider the statistic

$$T_j^{(1)} \triangleq \left\langle \sqrt{n} A_j, \frac{r_0}{\|r_0\|} \right\rangle, \quad 1 \le j \le M. \tag{7.4}$$

Since $r_0 = s$ is independent of each A_j, by Lemma 7.1, the random variables $T_j^{(1)}$, $1 \le j \le M$ are i.i.d. $N(0,1)$. Hence

$$\max_{1 \le j \le M} T_j^{(1)} = \left\langle \sqrt{n} A_{m_1}, \frac{r_0}{\|r_0\|} \right\rangle \approx \sqrt{2 \log M}. \tag{7.5}$$

The normalized norm of the residual $r_1 = r_0 - c_1 A_{m_1}$ is

$$
\begin{aligned}
|r_1|^2 &= |r_0|^2 + \frac{c_1^2}{n} |A_{m_1}|^2 - \frac{2 c_1 \|r_0\|}{n} \left\langle A_{m_1}, \frac{r_0}{\|r_0\|} \right\rangle \\
&\stackrel{(a)}{\approx} |r_0|^2 + \frac{c_1^2}{n} - \frac{2 c_1}{n} \frac{\|r_0\|}{\sqrt{n}} \sqrt{2 \log M} \\
&\stackrel{(b)}{\approx} \sigma^2 + \frac{c_1^2}{n} - \frac{2 c_1 \sigma}{n} \sqrt{2 \log M} \stackrel{(c)}{=} \sigma^2 \left(1 - \frac{2R}{L} \right).
\end{aligned}
\tag{7.6}
$$

Here (a) and (b) follow from (7.5) and the observations listed at the beginning of this section, while (c) follows by substituting for c_1 from (7.1) and using $n = L \log M / R$.

Step ℓ, $\ell = 2, \ldots, L$: We show that if $|r_{\ell-1}|^2 \approx \sigma^2 \left(1 - \frac{2R}{L}\right)^{\ell-1}$, then

$$|r_\ell|^2 \approx \sigma^2 \left(1 - \frac{2R}{L} \right)^\ell. \tag{7.7}$$

We already showed that (7.7) is true for $\ell = 1$.

For each $j \in \{(\ell-1)M + 1, \ldots, \ell M\}$, consider the statistic

$$T_j^{(\ell)} \triangleq \left\langle \sqrt{n} A_j, \frac{r_{\ell-1}}{\|r_{\ell-1}\|} \right\rangle. \tag{7.8}$$

109

Note that $r_{\ell-1}$ is independent of A_j because $r_{\ell-1}$ is a *function* of the source sequence s and the columns $\{A_j\}$, $1 \leq j \leq (\ell-1)M$, which are all independent of A_j for $j \in \{(\ell-1)M+1, \ldots, \ell M\}$. Therefore, by Lemma 7.1, the $T_j^{(\ell)}$'s are i.i.d. $\mathcal{N}(0,1)$ random variables for $j \in \{(\ell-1)M+1, \ldots, \ell M\}$. Hence, we have

$$\max_{(\ell-1)M+1 \leq j \leq \ell M} T_j^{(\ell)} = \left\langle \sqrt{n} A_{m_\ell}, \frac{r_{\ell-1}}{\|r_{\ell-1}\|} \right\rangle \approx \sqrt{2 \log M}. \tag{7.9}$$

From the expression for r_ℓ in (7.3), we have

$$\begin{aligned}
|r_\ell|^2 &= |r_{\ell-1}|^2 + \frac{c_\ell^2}{n} |A_{m_\ell}|^2 - \frac{2c_\ell}{n} \frac{\|r_{\ell-1}\|}{\sqrt{n}} \left\langle \sqrt{n} A_{m_\ell}, \frac{r_{\ell-1}}{\|r_{\ell-1}\|} \right\rangle \\
&\stackrel{(a)}{\approx} |r_{\ell-1}|^2 + \frac{c_\ell^2}{n} - \frac{2c_\ell |r_{\ell-1}|}{n} \sqrt{2 \log M} \\
&\stackrel{(b)}{\approx} \sigma^2 \left(1 - \frac{2R}{L}\right)^{\ell-1} + \frac{c_\ell^2}{n} - \frac{2c_\ell \sigma}{n} \left(1 - \frac{2R}{L}\right)^{(\ell-1)/2} \sqrt{2 \log M} \\
&\stackrel{(c)}{=} \sigma^2 \left(1 - \frac{2R}{L}\right)^\ell.
\end{aligned} \tag{7.10}$$

For (a) and (b) we have used (7.9) and the induction assumption on $|r_{\ell-1}|$. The equality (c) is obtained by substituting for c_ℓ from (7.1) and for n from (1.2). It can be verified that the chosen value of c_ℓ minimizes the third line in (7.10).

Therefore, when the algorithm terminates the final residual satisfies

$$|r_L|^2 = \left|s - A\hat{\beta}\right|^2 \approx \sigma^2 \left(1 - \frac{2R}{L}\right)^L \leq \sigma^2 e^{-2R} \tag{7.11}$$

where we have used the inequality $(1+x) \leq e^x$ for $x \in \mathbb{R}$.

Thus the encoding algorithm picks a codeword $\hat{\beta}$ that yields squared-error distortion approximately equal to $\sigma^2 e^{-2R}$, the Gaussian distortion-rate function at rate R. The heuristic analysis above is made rigorous (in the proof of Theorem 7.1) by bounding the deviation of the residual distortion each stage from its typical value.

7.3 Main result

Theorem 7.1. *Consider a length n source sequence s generated by an ergodic source with mean 0 and variance σ^2. Let $\delta_0, \delta_1, \delta_2$ be any positive constants such that*

$$\Delta \triangleq \delta_0 + 5R(\delta_1 + \delta_2) < \frac{1}{2}. \tag{7.12}$$

Let A be an $n \times ML$ design matrix with i.i.d. $\mathcal{N}(0, 1/n)$ entries and M, L satisfying (1.2). With the SPARC defined by A, the proposed encoding algorithm produces a codeword $A\hat{\beta}$ that satisfies the following for sufficiently large M, L.

$$P\left(\left|s - A\hat{\beta}\right|^2 > \sigma^2 e^{-2R}(1 + e^R \Delta)^2 \right) < p_0 + p_1 + p_2 \tag{7.13}$$

110

where

$$p_0 = P\left(\left|\frac{\|s\|}{\sigma} - 1\right| > \delta_0\right), \quad p_1 = 2ML\exp\left(-n\delta_1^2/8\right),$$

$$p_2 = \left(\frac{8\log M}{M^{2\delta_2}}\right)^L.$$

(7.14)

Remark 7.1. *For a given rate R, Theorem 7.1 guarantees that with high probability, the proposed encoder achieves distortion close to $D^*(R) = \sigma^2 e^{-2R}$ for any ergodic sources with variance σ^2. This complements the result in Theorem 6.2 for minimum-distance encoding.*

Corollary 7.2. *Let $\{\mathcal{S}_n\}_{n\geq 1}$ be a sequence of rate R SPARCs, indexed by block length n, with $M = L^b$, for $b > 0$. Then, for an i.i.d. $\mathcal{N}(0, \sigma^2)$ source, the sequence $\{\mathcal{S}_n\}_{n\geq 1}$ attains the optimal distortion-rate function $D^*(R) = \sigma^2 e^{-2R}$ with the proposed encoder. Furthermore, for any fixed distortion-level above $D^*(R)$, the probability of excess distortion decays exponentially with the block length n for sufficiently large n.*

Proof. For a fixed distortion-level $\sigma^2 e^{-2R} + \gamma$ with $\gamma > 0$, we can find $\Delta > 0$ such that $\sigma^2 e^{-2R} + \gamma = \sigma^2 e^{-2R}(1 + e^R\Delta)^2$. Equivalently, $\Delta > 0$ satisfies

$$\gamma = \sigma^2\Delta^2 + 2\Delta e^R\sigma^2.$$

(7.15)

Without loss of generality, we may assume that γ is small enough that Δ satisfying (7.15) lies in the interval $(0, \frac{1}{2})$. For positive constants $\delta_0, \delta_1, \delta_2$ chosen to satisfy (7.12), Theorem 7.1 implies that

$$P\left(\left|s - A\hat{\beta}\right|^2 > \sigma^2 e^{-2R} + \gamma\right) < p_0 + p_1 + p_2.$$

(7.16)

We now obtain upper bounds for p_0, p_1, p_2.

For an i.i.d. $\mathcal{N}(0, \sigma^2)$ source, $\|S\|^2/\sigma^2$ is a χ_n^2 random variable. A standard Chernoff bound yields

$$p_0 < 2\exp(-3n\delta_0^2/4).$$

(7.17)

Since $ML = L^{b+1}$ grows polynomially in n, the term p_1 in (7.14) can be expressed as

$$p_1 = \exp\left(-n\left(\frac{\delta_1^2}{8} - O(\frac{\log n}{n})\right)\right).$$

(7.18)

Finally using $M = L^b = \Theta((n/\log n)^b)$, we have

$$p_2 = \exp\left(-n\left(2\delta_2 R - O\left(\frac{\log\log n}{\log n}\right)\right)\right).$$

(7.19)

Using (7.17), (7.18) and (7.19) in (7.16), we conclude that for any fixed distortion-level $D^*(R) + \gamma$, the probability of excess distortion decays exponentially in n when n is sufficiently large. $\qquad\square$

7.3.1 Gap from $D^*(R)$

For a fixed R, to achieve distortions close to the optimal distortion-rate function $D^*(R) = \sigma^2 e^{-2R}$, we need p_0, p_1, p_2 to all go to 0. Ergodicity of the source ensures that that $p_0 \to 0$ as $n \to \infty$ (at a rate depending only on the source distribution). For p_2 to tend to 0 with growing L, from (7.14) we require that $M^{2\delta_2} > 8 \log M$. Or,

$$\delta_2 > \frac{\log \log M}{2 \log M} + \frac{\log 8}{2 \log M}. \tag{7.20}$$

To approach $D^*(R)$, we need n, L, M to all go to ∞ while satisfying (1.2): n, L need to be large for the probability of error in (7.14) to be small, while M needs to be large in order to allow δ_2 to be small according to (7.20).

When $M = L^b$, both L, M grow polynomially in n, and (7.20) implies that the gap from the optimal distortion $D^*(R)$ is $\Theta\left(\frac{\log \log n}{\log n}\right)$. On the other hand, if we choose $M = \kappa \log n$ for $\kappa > 0$, we have $L = \frac{nR}{\log(\kappa \log n)}$. In this case, the gap δ_2 from (7.20) is approximately $\frac{\log \log \log n}{\log \log n}$, i.e., the convergence to $D^*(R)$ with n is much slower. However, the per-sample computational complexity is $\Theta\left(\frac{n \log n}{\log \log n}\right)$, lower than the previous case, where the per-sample complexity was $\Theta\left((n/\log n)^{b+1}\right)$.

At the other extreme, $L = 1, M = e^{nR}$ reduces to the Shannon-style random codebook with. In this case, the SPARC consists of only one section and the proposed algorithm is essentially minimum-distance encoding. The computational complexity is $O(e^{nR})$, while the gap δ_2 from (7.20) is approximately $\frac{\log n}{n}$. The gap Δ from $D^*(R)$ is now dominated by δ_0 and δ_1 which are $\Theta(1/\sqrt{n})$, consistent with the results in [98, 61, 71].[1]

An interesting direction for future work is to design encoding algorithms with faster convergence to $D^*(R)$ while still having complexity that is polynomial in n.

7.3.2 Successive refinement interpretation

The encoding algorithm may be interpreted in terms of successive refinement source coding [38, 94]. We can think of each section of the design matrix A as a lossy codebook of rate R/L. For each section ℓ, $i = 1, \ldots, L$, the residual $r_{\ell-1}$ acts as the 'source' sequence, and the algorithm attempts to find the column *within* the section that minimizes the distortion. The distortion after section ℓ is the variance of the residual r_ℓ; this residual acts as the source sequence for section $\ell - 1$. Recall that the minimum mean-squared distortion achievable with a Gaussian codebook at rate R/L is [77]

$$D_\ell^* = |r_{\ell-1}|^2 \exp(-2R/L) \approx |r_{\ell-1}|^2 \left(1 - \frac{2R}{L}\right), \quad \text{for } R/L \ll 1. \tag{7.21}$$

---

[1] For $L = 1$, the factor ML that multiplies the exponential term in p_2 can be eliminated via a sharper analysis.

This minimum distortion can be attained with a codebook with elements chosen $\sim_{i.i.d.} \mathcal{N}(0, |r_{\ell-1}|^2 - D_\ell^*)$. From (7.1), recall that the codeword variance in section i of the codebook is

$$c_\ell^2 = \frac{2R\sigma^2}{L} \left(1 - \frac{2R}{L}\right)^{\ell-1} \approx |r_{\ell-1}|^2 - D_\ell^*, \tag{7.22}$$

where the approximate equality follows from (7.21) and (7.7). Therefore, the typical value of the distortion in Section i is close to D_ℓ^* since the algorithm is equivalent to minimum-distance encoding within each section. However, since the rate R/L is infinitesimal, the deviations from D_ℓ^* in each section can be significant. Despite this, when the number of sections L is large, Theorem 7.1 guarantees that the final distortion $|r_L^2|$ is close to the typical value $\sigma^2 e^{-2R}$.

A similar successive refinement approach was used in [85] to construct a lossy compression scheme that shares some similarities with the successive cancellation encoder.

7.4   Simulation results

In this section, we examine the empirical rate-distortion performance of the encoder via numerical simulations. The top graph in Fig. 7.1 shows the performance on a unit variance i.i.d Gaussian source. The dictionary dimension is $n \times ML$ with $M = L^b$. The curves show the average distortion at various rates for $b = 2$ and $b = 3$. The average was obtained from 70 random trials at each rate. Following convention, rates are plotted in bits rather than nats. The value of L was increased with rate in order to keep the total computational complexity ($\propto nL^{b+1}$) similar across different rates. Recall from (1.2) that the block length is determined by

$$n = \frac{bL \log L}{R}.$$

For example, for the rates $1.082, 2.092, 3.102$ and 4.112 bits/sample, L was chosen to be $46, 66, 81$ and 97, respectively. The corresponding values for the block length are $n = 705, 573, 497, 468$ for $b = 3$, and $n = 470, 382, 331, 312$ for $b = 2$. The graph shows the reduction in distortion obtained by increasing b from 2 to 3. This reduction comes at the expense of an increase in computational complexity by a factor of L. Simulations were also performed for a unit variance Laplacian source. The resulting distortion-rate curve was virtually identical to Fig. 7.1, which is consistent with Theorem 7.1.

For the simulations, a slightly modified version of the algorithm in Section 7.1 was used: the column selected in each iteration was based on minimum distance from the residual, rather than on maximum correlation as in (7.2). That is,

$$m_\ell = \underset{j:\ (\ell-1)M < j\ \leq \ell M}{\operatorname{argmin}} \|r_{\ell-1} - c_\ell A_j\|^2. \tag{7.23}$$

Though the two rules are similar for large n (since $\|A_j\| \approx 1$ for all j), we found the distance-based rule to give slightly better empirical performance.

Gish and Pierce [49] showed that uniform quantizers with entropy coding are nearly optimal at high rates and that their distortion for a unit variance source is well-approximated by $\frac{\pi e}{6} e^{-2R}$. (R
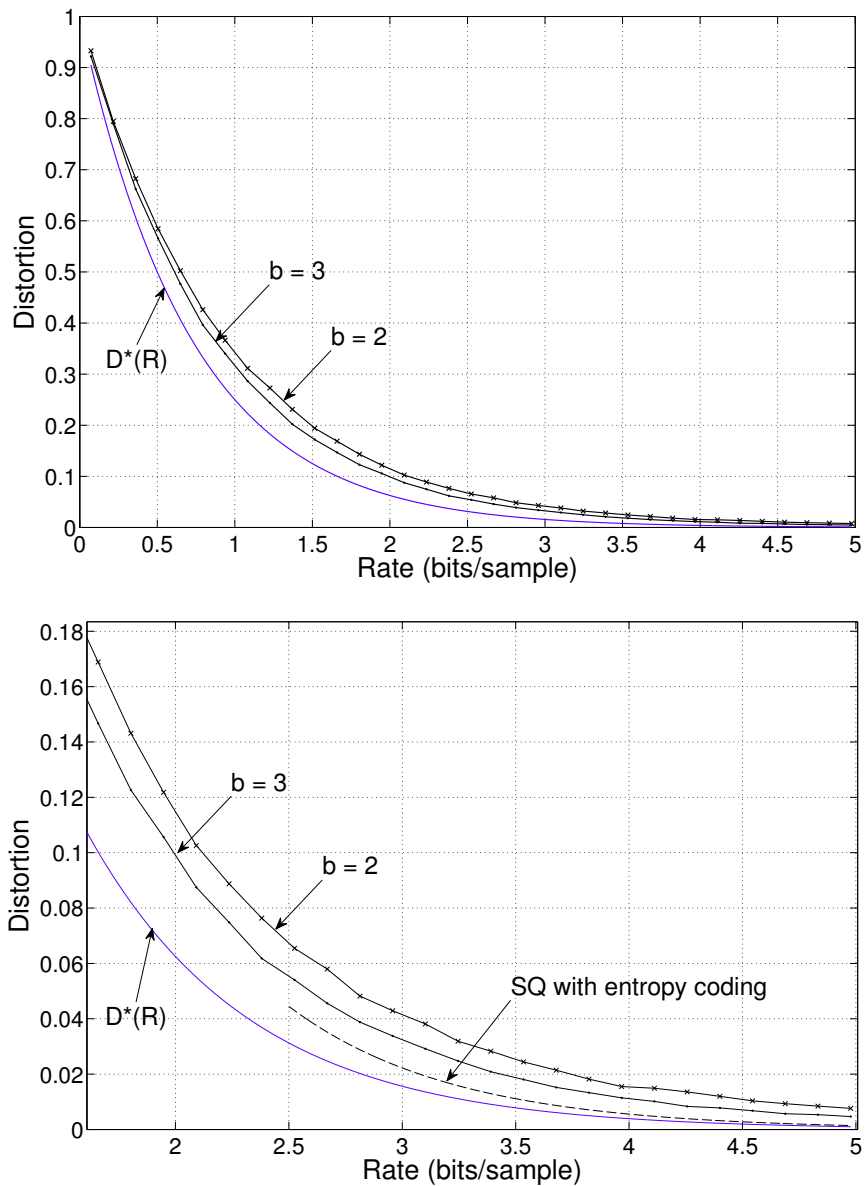
113

Figure 7.1: Top: Average distortion of the proposed encoder for i.i.d $\mathcal{N}(0,1)$ source. The design matrix has dimension $n \times ML$ with $M = L^b$. The distortion-rate performance is shown for $b = 2$ and $b = 3$ along with $D^*(R) = e^{-2R}$. Bottom: Focusing on the higher rates. The dashed line is the high-rate approximation for the distortion-rate function of an optimal entropy-coded scalar quantizer.

114

is the entropy of the quantizer in nats.) The bottom graph of Fig. 7.1 zooms in on the higher rates and shows the above high-rate approximation for the distortion of an optimal entropy-coded scalar quantizer (EC-SQ). Recall from (7.20) that the distortion gap from $D^*(R)$ is of the order of [2]

$$\delta_2 \approx \frac{\log \log M}{2 \log M} = \frac{\log b + \log \log L}{2b \log L},$$

which is comparable to the optimal $D^*(R) = e^{-2R}$ in the high-rate region. (In fact, δ_2 is larger than $D^*(R)$ at rates greater than 3 bits for the values of L and b we have used.) This explains the large ratio of the empirical distortion to $D^*(R)$ at higher rates.

In summary, the proposed encoder has good empirical performance, especially at low to moderate rates even with modest values of L and b. At high rates, there are a few other compression schemes including EC-SQs and the shape-gain quantizer of [57] whose empirical rate-distortion performance is close to optimal (see [57, Table III]).

7.5   Proof of Theorem 7.1

The proof involves analyzing the deviation from the typical values of the residual distortion at each step of the encoding algorithm. In particular, we have to deal with atypicality concerning the source sequence, the design matrix, and the maximum computed in each step of the algorithm.

We introduce some notation to capture the deviations from the typical values. Define Δ_0 via

$$|s|^2 = |r_0|^2 = \sigma^2 (1 + \Delta_0)^2. \tag{7.24}$$

The deviation of the norm of the residual at stage $i = 1, \ldots, L$ from its typical value is captured by Δ_i, defined via

$$|r_i|^2 = \sigma^2 \left(1 - \frac{2R}{L}\right)^i (1 + \Delta_i)^2. \tag{7.25}$$

The deviation in the norm of A_{m_i} (the column chosen in step i) is captured by γ_i, defiend as

$$|A_{m_i}|^2 = 1 + \gamma_i, \quad i = 1, \ldots, L. \tag{7.26}$$

Recall that the statistics $T_j^{(i)}$ defined in (7.8) are i.i.d $\mathcal{N}(0,1)$ for $j \in \{(i-1)M + 1, \ldots, iM\}$. We write

$$\max_{(i-1)M+1 \leq j \leq iM} T_j^{(i)} = \left\langle A_{m_i}, \frac{r_{i-1}}{\|r_{i-1}\|} \right\rangle$$
$$= \sqrt{2 \log M}(1 + \epsilon_i), \quad i = 1, \ldots, L. \tag{7.27}$$

The ϵ_i measure the deviations of the maximum from $\sqrt{2 \log M}$ in each step.

---

[2]The constants in Theorem 7.1 are not optimized, so the theorem does not give a very precise estimate of the excess distortion in the high-rate, low-distortion regime.

With this notation, using the expression for r_i from (7.3) we have

$$
\begin{aligned}
|r_i|^2 &= \sigma^2 \left(1 - \frac{2R}{L}\right)^i (1 + \Delta_i)^2 \\
&= |r_{i-1}|^2 + c_i^2 |A_{m_i}|^2 - \frac{2c_i \|r_{i-1}\|}{n} \left\langle A_{m_i}, \frac{r_{i-1}}{\|r_{i-1}\|} \right\rangle \\
&= \sigma^2 \left(1 - \frac{2R}{L}\right)^{i-1} (1 + \Delta_{i-1})^2 + c_i^2 (1 + \gamma_i) \\
&\quad - 2c_i \sigma \left(1 - \frac{2R}{L}\right)^{\frac{i-1}{2}} (1 + \Delta_{i-1}) \sqrt{\frac{2 \log M}{n}} (1 + \epsilon_i) \\
&= \sigma^2 \left(1 - \frac{2R}{L}\right)^i \left((1 + \Delta_{i-1})^2 + \frac{\frac{2R}{L}}{1 - \frac{2R}{L}} (\Delta_{i-1}^2 + \gamma_i - 2\epsilon_i(1 + \Delta_{i-1})) \right).
\end{aligned}
\tag{7.28}
$$

From (7.28), we obtain

$$
(1 + \Delta_i)^2 = (1 + \Delta_{i-1})^2 + \frac{\frac{2R}{L}}{1 - \frac{2R}{L}} (\Delta_{i-1}^2 + \gamma_i - 2\epsilon_i(1 + \Delta_{i-1})), \quad i \in [L].
\tag{7.29}
$$

The goal is to bound the final distortion given by

$$
|r_L|^2 = \sigma^2 \left(1 - \frac{2R}{L}\right)^L (1 + \Delta_L)^2.
\tag{7.30}
$$

We would like to find an upper bound for $(1 + \Delta_L)^2$ that holds under an event whose probability is close to 1. Accordingly, define $\mathcal{A}$ as the event where *all* of the following hold:

1. $|\Delta_0| < \delta_0$,

2. $\sum_{i=1}^{L} \frac{|\gamma_i|}{L} < \delta_1$,

3. $\sum_{i=1}^{L} \frac{|\epsilon_i|}{L} < \delta_2$,

for $\delta_0, \delta_1, \delta_2$ that satisfy (7.12). We upper bound the probability of the event $\mathcal{A}^c$ using the following large deviations bounds.

Lemma 7.3. *For $\delta \in (0, 1]$, $P\left(\frac{1}{L} \sum_{i=1}^{L} |\gamma_i| > \delta\right) < 2ML \exp\left(-n\delta^2/8\right).$*

Lemma 7.4. *For $\delta > 0$, $P\left(\frac{1}{L} \sum_{i=1}^{L} |\epsilon_i| > \delta\right) < \left(\frac{M^{2\delta}}{8 \log M}\right)^{-L}.$*

The proofs of these lemmas can be found in Appendix II and III of [113], respectively.

Using these lemmas, we have

$$
P(\mathcal{A}^c) < p_0 + p_1 + p_2
\tag{7.31}
$$

where p_0, p_1, p_2 are given by (7.14). The remainder of the proof consists of obtaining a bound for $(1 + \Delta_L)^2$ under the condition that $\mathcal{A}$ holds.

116

Lemma 7.5. *For all sufficiently large L, when $\mathcal{A}$ holds we have*

$$\Delta_i \geq \Delta_0 - \frac{4R}{1 - 2R/L} \left(\sum_{j=1}^{i} \frac{|\gamma_j| + |\epsilon_j|}{L} \right), \quad i = 1, \ldots, L. \tag{7.32}$$

In particular, $\Delta_i > -\frac{1}{2}, \quad i = 1, \ldots, L$

Proof. We first show that $\Delta_i > -\frac{1}{2}$ follows from (7.32). Indeed, (7.32) implies that

$$\Delta_i \geq \Delta_0 - \frac{4R}{1 - 2R/L} \left(\sum_{j=1}^{i} \frac{|\gamma_j| + |\epsilon_j|}{L} \right) \overset{(a)}{>} -\delta_0 - 5R \left(\delta_1 + \delta_2 \right) \overset{(b)}{>} -\frac{1}{2} \tag{7.33}$$

where (a) is obtained from the conditions of $\mathcal{A}$ while (b) holds due to (7.12).

The statement (7.32) trivially holds for $i = 0$. Towards induction, assume (7.32) holds for $i - 1$ for some $i \in \{1, \ldots, L\}$. From (7.29), we obtain

$$(1 + \Delta_i)^2 = (1 + \Delta_{i-1})^2 + \frac{2R/L}{1 - 2R/L} (\Delta_{i-1}^2 + \gamma_i - 2\epsilon_i(1 + \Delta_{i-1}))$$

$$\geq (1 + \Delta_{i-1})^2 - \frac{2R/L}{1 - 2R/L} (|\gamma_i| + 2|\epsilon_i|(1 + \Delta_{i-1})). \tag{7.34}$$

For L large enough, the right side above is positive and we therefore have

$$(1 + \Delta_i) \geq (1 + \Delta_{i-1}) \left[1 - \frac{2R/L}{1 - 2R/L} \left[\frac{|\gamma_i|}{(1 + \Delta_{i-1})^2} + \frac{2|\epsilon_i|}{1 + \Delta_{i-1}} \right] \right]^{\frac{1}{2}}$$

$$\geq 1 + \Delta_{i-1} - \frac{2R/L}{1 - 2R/L} \left(\frac{|\gamma_i|}{(1 + \Delta_{i-1})} + 2|\epsilon_i| \right), \tag{7.35}$$

where the second inequality is obtained using $\sqrt{1 - x} \geq 1 - x$ for $x \in (0, 1)$. We therefore have

$$\Delta_i \geq \Delta_{i-1} - \frac{2R/L}{1 - 2R/L} \left(\frac{|\gamma_i|}{(1 + \Delta_{i-1})} + 2|\epsilon_i| \right)$$

$$\overset{(a)}{\geq} \Delta_{i-1} - \frac{2R/L}{1 - 2R/L} (2|\gamma_i| + 2|\epsilon_i|) \tag{7.36}$$

$$\overset{(b)}{\geq} \Delta_0 - \frac{4R}{1 - 2R/L} \left(\sum_{j=1}^{i-1} \frac{|\gamma_j| + |\epsilon_j|}{L} \right) - \frac{4R/L}{1 - 2R/L} (|\gamma_i| + |\epsilon_i|).$$

In the chain above, (a) holds because $\Delta_{i-1} > \frac{1}{2}$, a consequence of the induction hypothesis as shown in (7.33). (b) is obtained by using the induction hypothesis for Δ_{i-1}. $\qquad\square$

Lemma 7.6. *When $\mathcal{A}$ is true and L is large enough that Lemma 7.5 holds,*

$$|\Delta_i| \leq |\Delta_0| w^i + \frac{4R/L}{1 - 2R/L} \sum_{j=1}^{i} w^{i-j} (|\gamma_j| + |\epsilon_j|) \tag{7.37}$$

for $i = 1, \ldots, L$, where $w = \left(1 + \frac{R/L}{1 - 2R/L} \right)$.

117

Proof. We prove the lemma by induction. For $i = 1$, we have from (7.29)

$$(1 + \Delta_1)^2 = (1 + \Delta_0)^2 + \frac{\frac{2R}{L}}{1 - \frac{2R}{L}} (\Delta_0^2 + \gamma_1 - 2\epsilon_1(1 + \Delta_0))$$

$$= (1 + |\Delta_0|)^2 \left[1 + \frac{\frac{2R}{L}}{1 - \frac{2R}{L}} \left(\frac{\Delta_0^2}{(1 + |\Delta_0|)^2} + \frac{|\gamma_1|}{(1 + |\Delta_0|)^2} + \frac{2|\epsilon_1|}{(1 + |\Delta_0|)} \right) \right].$$

(7.38)

Therefore,

$$1 + \Delta_1 \le (1 + |\Delta_0|) \left[1 + \frac{\frac{2R}{L}}{1 - \frac{2R}{L}} \left(\frac{\Delta_0^2}{(1 + |\Delta_0|)^2} + \frac{|\gamma_1|}{(1 + |\Delta_0|)^2} + \frac{2|\epsilon_1|}{(1 + |\Delta_0|)} \right) \right]^{\frac{1}{2}}$$

$$\le (1 + |\Delta_0|) \left[1 + \frac{\frac{R}{L}}{1 - \frac{2R}{L}} \left(\frac{\Delta_0^2}{(1 + |\Delta_0|)^2} + \frac{|\gamma_1|}{(1 + |\Delta_0|)^2} + \frac{2|\epsilon_1|}{(1 + |\Delta_0|)} \right) \right]$$

(7.39)

where we have used the inequality $\sqrt{1 + x} \le 1 + \frac{x}{2}$ for $x > 0$. We therefore have

$$\Delta_1 \le |\Delta_0| + \frac{R/L}{1 - 2R/L} \left(\frac{\Delta_0^2}{(1 + |\Delta_0|)} + \frac{|\gamma_1|}{(1 + |\Delta_0|)} + 2|\epsilon_1| \right)$$

$$\overset{(a)}{\le} |\Delta_0| + \frac{R/L}{1 - 2R/L} (|\Delta_0| + |\gamma_1| + 2|\epsilon_1|)$$

$$\le |\Delta_0| \left(1 + \frac{R/L}{1 - 2R/L} \right) + \frac{2R/L}{1 - 2R/L} (|\gamma_1| + |\epsilon_1|),$$

(7.40)

where (a) is obtained using $|\Delta_0| / (1 + |\Delta_0|) < 1$. From Lemma 7.5, we have

$$\Delta_1 \ge \Delta_0 - \frac{4R/L}{1 - 2R/L} (|\gamma_1| + |\epsilon_1|)$$

$$\ge -|\Delta_0| - \frac{4R/L}{1 - 2R/L} (|\gamma_1| + |\epsilon_1|).$$

(7.41)

Combining (7.40) and (7.41), we obtain

$$|\Delta_1| \le |\Delta_0| \left(1 + \frac{R/L}{1 - 2R/L} \right) + \frac{4R/L}{1 - 2R/L} (|\gamma_1| + |\epsilon_1|).$$

(7.42)

This completes the proof for $i = 1$.

Towards induction, assume that the lemma holds for $i - 1$. From (7.29), we obtain

$$(1 + \Delta_i)^2 \le 1 + \Delta_{i-1}^2 + 2|\Delta_{i-1}|$$

$$+ \frac{2R/L}{1 - 2R/L} (\Delta_{i-1}^2 + |\gamma_i| + 2|\epsilon_i| (1 + |\Delta_{i-1}|)).$$

(7.43)

Using arguments identical to those in (7.38)–(7.40), we get

$$\Delta_i \le |\Delta_{i-1}| \left(1 + \frac{R/L}{1 - 2R/L} \right) + \frac{2R/L}{1 - 2R/L} (|\gamma_i| + |\epsilon_i|).$$

(7.44)

From the proof of Lemma 7.5 (see (7.36)), we have

$$\Delta_i \geq \Delta_{i-1} - \frac{4R/L}{1 - 2R/L}(|\gamma_i| + |\epsilon_i|)$$

$$\geq -|\Delta_{i-1}| - \frac{4R/L}{1 - 2R/L}(|\gamma_i| + |\epsilon_i|). \tag{7.45}$$

Combining (7.44) and (7.45), we obtain

$$|\Delta_i| \leq |\Delta_{i-1}|\left(1 + \frac{R/L}{1 - 2R/L}\right) + \frac{4R/L}{1 - 2R/L}(|\gamma_i| + |\epsilon_i|). \tag{7.46}$$

Using the induction hypothesis to bound $|\Delta_{i-1}|$ in (7.46), we obtain

$$|\Delta_i| \leq \left(|\Delta_0|\, w^{i-1} + \frac{4R/L}{1 - 2R/L}\sum_{j=1}^{i-1} w^{i-1-j}(|\gamma_j| + |\epsilon_j|)\right)$$

$$\cdot \left(1 + \frac{R/L}{1 - 2R/L}\right) + \frac{4R/L}{1 - 2R/L}(|\gamma_i| + |\epsilon_i|)$$

$$= |\Delta_0|\, w^i + \frac{4R/L}{1 - 2R/L}\sum_{j=1}^{i} w^{i-j}(|\gamma_j| + |\epsilon_j|),$$

as required. $\qquad\qquad\square$

Lemma 7.6 implies that when $\mathcal{A}$ holds and L is sufficiently large,

$$|\Delta_L| \leq |\Delta_0|\, w^L + \frac{4R/L}{1 - 2R/L}\sum_{j=1}^{L} w^{L-j}(|\gamma_j| + |\epsilon_j|)$$

$$\leq w^L\left[|\Delta_0| + \frac{4R}{(1 - 2R/L)w}\left(\sum_{j=1}^{L} \frac{|\gamma_j|}{L} + \sum_{j=1}^{L} \frac{|\epsilon_j|}{L}\right)\right]$$

$$\overset{(a)}{\leq} w^L\left[\delta_0 + \frac{4R}{(1 - R/L)}(\delta_1 + \delta_2)\right] \tag{7.47}$$

$$\overset{(b)}{\leq} \exp\left(\frac{R}{1 - 2R/L}\right)\left[\delta_0 + \frac{4R}{(1 - R/L)}(\delta_1 + \delta_2)\right]$$

$$\leq e^R\left(\delta_0 + 5R(\delta_1 + \delta_2)\right) \quad \text{for large enough } L.$$

In the above chain, (a) is true because $\mathcal{A}$ holds, and (b) is obtained by applying the inequality $1 + x \leq e^x$ with $x = \frac{R/L}{1 - 2R/L}$.

Hence when $\mathcal{A}$ holds and L is sufficiently large, the distortion can be bounded as

$$|R_L|^2 = \sigma^2 e^{-2R}(1 + \Delta_L)^2 \leq \sigma^2 e^{-2R}(1 + |\Delta_L|)^2$$

$$\overset{(c)}{\leq} \sigma^2 e^{-2R}(1 + e^R\Delta)^2 \tag{7.48}$$

where (c) follows from (7.47) by defining $\Delta = \delta_0 + 5R(\delta_1 + \delta_2)$. Combining (7.48) with (7.31) completes the proof of the theorem.

119

Part III

Multiuser Communication and Compression with SPARCs

Chapter 8

Broadcast and Multiple-access Channels

In the final part of the monograph, we discuss the use of SPARCs for multiuser channel and source coding models. It is well known [37] that the optimal rate regions for several multiuser channel and source coding problems can be achieved using the following ingredients: i) rate-optimal point-to-point source and channel codes, and ii) combining or splitting these point-to-point codes via superposition or random binning. In this chapter, we show how superposition coding can be implemented using SPARCs for Gaussian broadcast and multiple-access channels. In the next chapter, we describe how random binning can be implemented using SPARCs.

All rates within the capacity region of the Gaussian broadcast and multiple access channels can be achieved by combining codes designed for point-to-point Gaussian channels [32, 37]. Therefore the SPARC construction for point-to-point channels, where codewords are defined as the superposition of columns of a matrix, can be easily extended to these multiuser channels.

8.1   The Gaussian broadcast channel

The K-user AWGN broadcast channel has a single transmitter and K output sequences, one for each user. The input sequence $x = (x_1, \ldots, x_n)$ transmitted over the broadcast channel has to satisfy an average power constraint: $\frac{1}{n} \sum_j x_j^2 \leq P$. The channel output sequence of user $i \in [K]$ is denoted by $y^{(i)} = (y_1^{(i)}, \ldots, y_n^{(i)})$, where the jth output symbol is produced as $y_j^{(i)} = x_j + w_j^{(i)}$, for $j \in [n]$. The noise variables $(w_j^{(i)})_{j \in [n]}$ are i.i.d. $\sim \mathcal{N}(0, \sigma_i^2)$, where σ_i^2 is the noise variance of the ith receiver.

We will focus on the two-user broadcast channel for simplicity, although the coding schemes can be extended to $K > 2$ users in a straightforward way. Throughout, we will assume $\sigma_1^2 < \sigma_2^2$, i.e., the noise at the first receiver has a lower variance than the noise at the second.

If we denote the rates of the two users by R_1 and R_2, then the capacity region [37, Chapter 5] is
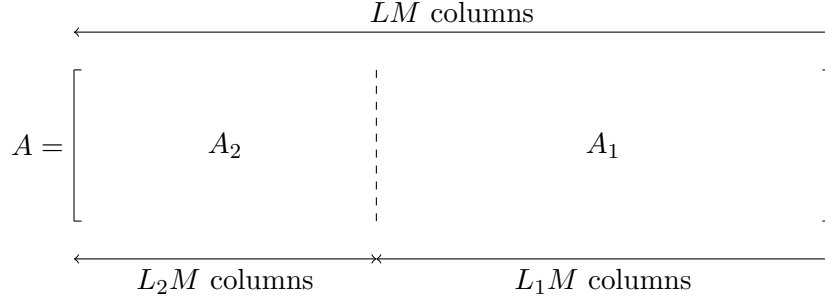
Figure 8.1: Division of the SPARC design matrix A for two users in the Gaussian broadcast channel. The second user is allocated the first L_2 sections, and the first user is allocated the remaining L_1 section.

the union of all rate pairs (R_1, R_2) over $\alpha \in [0, 1]$ which satisfy

$$R_1 \leq \frac{1}{2} \log \left(1 + \frac{\alpha P}{\sigma_1^2} \right), \tag{8.1}$$

$$R_2 \leq \frac{1}{2} \log \left(1 + \frac{(1-\alpha)P}{\alpha P + \sigma_2^2} \right). \tag{8.2}$$

It is well known that any rate pair within the capacity region can be achieved using superposition coding [31]. In superposition coding, the transmitted codeword x is generated as the sum of two independent codewords $x^{(1)}, x^{(2)}$, with powers $\alpha P, (1-\alpha)P$, drawn from codebooks of size 2^{nR_1} and 2^{nR_2}, respectively.

Receiver 2 has to decode $x^{(2)}$ from its output sequence $y^{(2)} = x^{(2)} + x^{(1)} + w^{(2)}$. Treating $x^{(1)}$ as interference (with average power αP), receiver 2 has an effective point-to-point channel with signal to noise ratio $\frac{(1-\alpha)P}{\alpha P + \sigma_2^2}$. If receiver 2 can reliably decode $x^{(2)}$, receiver 1 will be also able to first decode $x^{(2)}$ (with high probability) from $y^{(1)} = x^{(2)} + x^{(1)} + w^{(1)}$. This is because $\sigma_1^2 \leq \sigma_2^2$. After subtracting the decoded $x^{(2)}$ from $y^{(1)}$, receiver 1 has an effective point-to-point channel with $\mathsf{snr} = \frac{\alpha P}{\sigma^2}$ to decoder $x^{(1)}$.

We now implement this superposition coding scheme with SPARCs.

8.2 SPARCs for the Gaussian broadcast channel

Fix rates R_1, R_2 that lie within the capacity region (8.1)–(8.2). The two users' codebooks are defined via SPARC design matrices A_1 and A_2 with parameters (n, M, L_1) and (n, M, L_2), respectively. The parameters are chosen such that

$$nR_1 = L_1 \log M, \qquad nR_2 = L_2 \log M. \tag{8.3}$$

The entries of A_1, A_2 are chosen $\sim_{\text{i.i.d}} \mathcal{N}(0, 1/n)$.

124

By concatenating the two design matrices, we obtain a SPARC defined by $A = [A_2 \ A_1]$ with $L = L_1 + L_2$ sections. This combined SPARC, shown in Figure 8.1, has rate $R_1 + R_2$, and from (8.3) we see that its parameters satisfy

$$n(R_1 + R_2) = L \log M.$$

Power allocation The non-zero coefficients in the sections of users 1 and 2 are set to $\{\sqrt{nP_{1\ell}}\}_{\ell \in [L_1]}$ and $\{\sqrt{nP_{2\ell}}\}_{\ell \in [L_2]}$, respectively. For optimal (ML) decoding, we use a flat power allocation, i.e., $P_{1\ell} = \sqrt{\frac{(1-\alpha)P}{L_1}}$ and $P_{2\ell} = \sqrt{\frac{\alpha P}{L_2}}$, respectively. For AMP decoding, the two power allocations $\{P_{1,\ell}\}_{\ell \in [L_1]}$ and $\{P_{2,\ell}\}_{\ell \in [L_2]}$ are chosen in the same way as for point-to-point SPARCs. For example, one could use the power allocation determined by the iterative algorithm in Chapter 4 using the parameters (L_i, R_i, P_i) for user $i \in \{1, 2\}$.

Encoding The message of each user $i \in \{1, 2\}$ mapped to a message vector $\beta^{(i)} \in \mathcal{B}_{M,L_i}$. The concatenated message vector is denoted by $\beta \in \mathcal{B}_{M,L}$. The transmitted codeword is

$$x = A\beta = [A_1 \quad A_2] \begin{bmatrix} \beta^{(1)} \\ \beta^{(2)} \end{bmatrix} = A_1 \beta^{(1)} + A_2 \beta^{(2)}.$$

Optimal decoding Receiver 2 (with the higher noise variance) decodes

$$\hat{\beta}^{(2)}_{\mathsf{opt}} = \underset{\hat{\beta}^{(2)} \in \mathcal{B}_{M,L_2}}{\arg\min} \ \|y^{(2)} - A_2 \hat{\beta}^{(2)}\|^2. \tag{8.4}$$

Receiver 1 first decodes the concatenated message vector β as

$$\hat{\beta}_{\mathsf{opt}} = \underset{\hat{\beta} \in \mathcal{B}_{M,L}}{\arg\min} \|y^{(1)} - A\hat{\beta}\|^2, \tag{8.5}$$

and then reconstructs $\beta^{(1)}_{\mathsf{opt}}$ by taking the last L_1 sections.

AMP decoding Receiver 2 decodes $\hat{\beta}^{(2)}$ by running a standard SPARC AMP decoding routine (as described in Section 3.4), using the design matrix A_2, i.e., the first L_2 columns in A.

Receiver 1 decodes $\hat{\beta}$ via AMP decoding on the concatenated design matrix A, with the combined power allocation $\{P_\ell\}_{\ell \in [L]}$ given by

$$P_\ell = \begin{cases} P_{2,\ell} & \ell \leq L_2 \\ P_{1,\ell-L_2} & L_2 < \ell \leq L = L_1 + L_2. \end{cases}$$

The last L_1 sections of $\hat{\beta}$ represent $\hat{\beta}^{(1)}$.

8.3 Bounds on error performance

8.3.1 Optimal decoding

As seen from (8.4) and (8.5), each receiver uses a point-to-point SPARC decoder, using design matrix A_2 for receiver 2 and A for receiver 1. Therefore, Theorem 2.1 can be directly applied to obtain bounds on the probability of excess section error rate at each receiver with optimal decoding.

Theorem 8.1. *Let* $M = L^{\mathsf{a}}$, *with* a *such that* $\mathsf{a} \geq \max\{\mathsf{a}_L^*(\mathsf{snr}), \mathsf{a}_{L_2}^*(\mathsf{snr})\}$, *where* $\mathsf{a}_L^*(\mathsf{snr})$ *is defined in (2.2). Let* (R_1, R_2) *be a rate pair within the capacity region given by (8.1)–(8.2). Let*

$$\Delta = \frac{1}{2}\log\left(1 + \frac{P}{\sigma_1^2}\right) - (R_1 + R_2), \quad \Delta_2 = \frac{1}{2}\log\left(1 + \frac{(1-\alpha)P}{\alpha P + \sigma_2^2}\right) - R_2 \tag{8.6}$$

be strictly positive distances (of $(R_1 + R_2)$ *and* R_2, *respectively) from a point on the boundary parametrized by some* $\alpha \in [0, 1]$. *Then with optimal decoding, for any* $\epsilon_1, \epsilon_2 > 0$ *the section error rates* $\mathcal{E}_{sec}^1$ *and* $\mathcal{E}_{sec}^2$ *at the two receivers satisfy*

$$\mathbb{P}\left(\mathcal{E}_{sec}^1 \geq \epsilon_1\right) = e^{-nE_1(\epsilon_1, \Delta)}, \quad \mathbb{P}\left(\mathcal{E}_{sec}^2 \geq \epsilon_2\right) = e^{-nE_2(\epsilon_2, \Delta_2)} \tag{8.7}$$

where

$$E_1(\epsilon_1, \Delta) \geq h(\epsilon_1, \Delta) - \frac{\log 2L}{n}, \quad E_2(\epsilon_2, \Delta_2) \geq h(\epsilon_2, \Delta_2) - \frac{\log 2L_2}{n},$$

where $h(\cdot, \cdot)$ *is defined in (2.20).*

Proof. The result follows by applying Theorem 2.1 to decoder 2 which performs point-to-point decoding on a rate R_2 SPARC defined by A_2, and to decoder 1 which decodes a rate $R_1 + R_2$ SPARC of defined by $A = [A_1 \ A_2]$. □

As in Proposition 2.2, one can bound the probabilities of message error, i.e., $P(\hat{\beta}^{(1)} \neq \beta^{(1)})$ and $P(\hat{\beta}^{(2)} \neq \beta^{(2)})$, by using an outer Reed-Solomon code. For any $\epsilon > 0$ and rate pair (R_1, R_2) inside the capacity region, by using an outer RS code of rate $(1 - 2\epsilon)$ for each of the component SPARCs, one obtains a code with rates $(R_1(1 - 2\epsilon), R_2(1 - 2\epsilon))$ with message error probabilities of the two users bounded by $e^{-nE_1(\epsilon, \Delta)}$ and $e^{-nE_2(\epsilon, \Delta_2)}$, respectively.

8.3.2 AMP decoding

For AMP decoding, we can apply Theorem 3.3 to obtain a bound on the probability of excess section error rate with an exponentially decaying allocation for each user. That is, with $P_1 = \alpha P$ and $P_2 = (1 - \alpha)P$, the power allocation for design matrix A_1 is

$$P_{1\ell} = \kappa_1 \left(1 + \frac{\alpha P}{\sigma_1^2}\right)^{-\ell/L_1}, \quad \ell \in [L_1]. \tag{8.8}$$

The power allocation for A_2 is

$$P_{2,\ell} = \kappa_2 \left(1 + \frac{(1-\alpha)P}{\alpha P + \sigma_2^2} \right)^{-\ell/L_2}, \quad \ell \in [L_2]. \tag{8.9}$$

Here κ_1, κ_2 are normalizing constants chosen to satisfy the two power constraints.

Theorem 8.2. *Fix any rate pair (R_1, R_2) within the capacity region (8.1)–(8.2). Consider a broadcast SPARC defined by design matrix $A = [A_1 \, A_2]$ with parameters n, M, L_1, L_2, that satisfy (8.3), and a power allocation given by (8.8) and (8.9).*

Fix $\epsilon > 0$. Then for sufficiently large L_1, L_2, M, the section error rate of the AMP decoder satisfies

$$P\left(\mathcal{E}_{sec}^1 > \epsilon \right) \le K_T \exp \left\{ \frac{-\kappa_T L}{(\log M)^{2T-1}} \left(\frac{\epsilon \sigma_1^2 \mathcal{C}}{2} - P f_R(M) \right)^2 \right\},$$

$$P\left(\mathcal{E}_{sec}^2 > \epsilon \right) \le K_T \exp \left\{ \frac{-\kappa_T L_2}{(\log M)^{2T-1}} \left(\frac{\epsilon \sigma_2^2 \mathcal{C}_2}{2} - P f_R(M) \right)^2 \right\},$$

$$\tag{8.10}$$

where $\mathcal{C} = \frac{1}{2} \log(1 + \frac{P}{\sigma_1^2})$, $\mathcal{C}_2 = \frac{1}{2} \log(1 + \frac{(1-\alpha)P}{\alpha P + \sigma_2^2})$, T is the maximum number of iterations of the two AMP decoders, and κ_T, K_T are defined in Theorem 3.3. Furthermore, T is inversely proportional to the minimum of Δ, Δ_2, which are defined in (8.6).

Proof. It can be verified (using Lemma 3.3) that the state evolution recursion (3.25) predicts reliable decoding in the large system limit for any rate pair within the capacity region and the specified power allocation. The result then follows using arguments very similar to those used to prove Theorem 3.3. $\square$

8.4   Simulation results

We now discuss the empirical performance of SPARCs for the Gaussian broadcast channel with AMP decoding, considering a setup with $P = 63$, $\sigma_1^2 = 1$, and $\sigma_2^2 = 2$. Each operating point is set as follows: fix $\alpha \in [0, 1]$, which specifies the balance of power between the two receivers. The point on the boundary of the capacity region corresponding to this α is $\mathcal{C}_1 = \frac{1}{2} \log(1 + \frac{\alpha P}{\sigma_1^2})$ and $\mathcal{C}_2 = \frac{1}{2} \log(1 + \frac{1-\alpha}{\alpha P + \sigma_2^2})$. Fix $\gamma \in [0, 1]$ and set $R_1 = \gamma \mathcal{C}_1$ and $R_2 = \gamma \mathcal{C}_2$. The parameter γ determines the back-off from the boundary point $(\mathcal{C}_1, \mathcal{C}_2)$ of the capacity region.

For the SPARC design matrix, we first fix $M = 512$ and $n = 4095$. The parameters L_1, L_2 are then determined by the rate pair as $L_1 = nR_1/\log M$, and $L_2 = nR_2/\log M$. With $P_1 = \alpha P$ and $P_2 = (1 - \alpha)P$, the value of the non-zero coefficient in each section is set using the iterative power allocation algorithm as discussed on p. 125.

The encoding and decoding operations are then performed as described in Section 8.2. Figure 8.2 shows the bit-error rate performance in three charts. The two charts on the top show the bit error rate performance achieved when only considering the first and second receiver, while the third chart
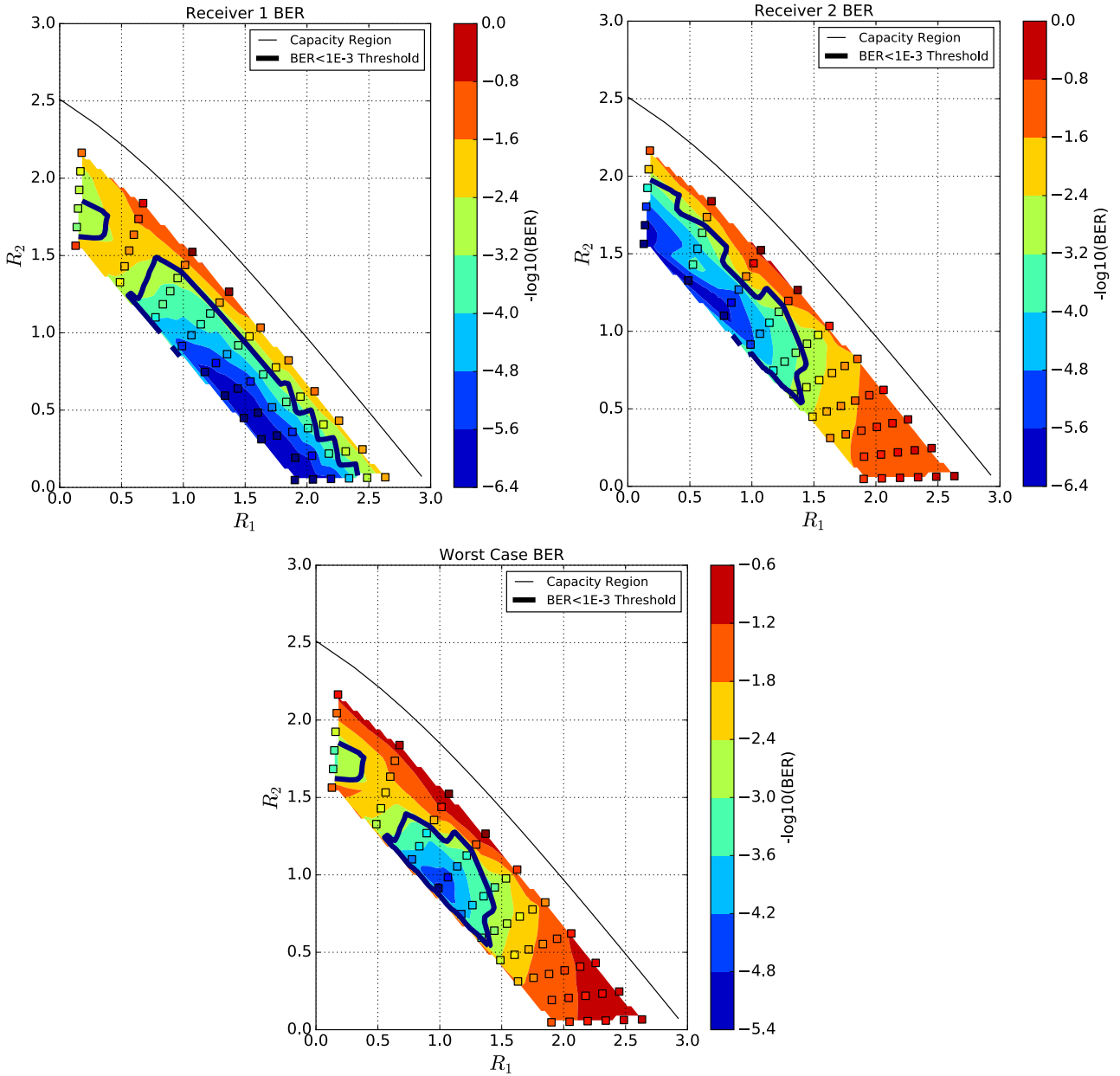
Figure 8.2: Bit error rates for the broadcast channel with AMP decoding, with contour indicating the boundary of the regions where the error rate is below 10^{-3}. Each displayed point is the average of approximately 3000 trials. $P = 63$, $\sigma_1^2 = 1.0$, $\sigma_2^2 = 2.0$, $M = 512$, $n = 4095$.

shows the worst case of those two. Additionally, a contour is plotted showing the boundaries of regions where the bit error rate is found to be below 10^{-3}.

When considering each receiver independently, we observe that the bit error rates are reasonably low when that receiver's rate is above 0.5, which is in accordance with the results from point-to-point channels. However, because each receiver suffers badly once its rate goes below 0.5, the worst-case error is only better than 10^{-3} in a small number of cases, relatively far from the capacity boundary and only near equal power balance where $\alpha = 0.5$.

Degradation of decoding performance at low rates is already observed in the simpler point-to-point case, but in the broadcast setup there is an additional factor contributing to the performance. For the experimental setup as described, M is fixed to the same value for both users. As we saw in Chapter 4, for a particular channel set-up, there is an optimal M, above and below which performance can rapidly decrease. Therefore, when the rates for the two receivers in the BC setup differ substantially, so too will the optimal M. The gap between each receiver's optimum M and the chosen value will lead to performance degradation, as observed. It is conceptually possible to run the AMP decoder with differing values for M in different sections, which would allow each receiver to operate on an optimal M, but this has not been explored.

8.5   The Gaussian multiple-access channel

The K-user Gaussian multiple-access channel (MAC) has K transmitters and a single receiver. The codeword transmitted by user $i \in [K]$, denoted by $x^{(i)} = (x_1^{(i)}, \ldots, x_n^{(i)})$, has to satisfy an average power constraint P_i. The channel output sequence is $y = \sum_{i=1}^{K} x^{(i)} + w$, where $w \in \mathbb{R}^n$ is $\sim_{\text{i.i.d.}} \mathcal{N}(0, \sigma^2)$.

We will focus on the two-user MAC for simplicity, although the coding schemes can be extended to $K > 2$ users in a straightforward manner. Denoting the rates of the two users by R_1 and R_2, the capacity region is the set of all rate pairs (R_1, R_2) which satisfy [37, Chapter 4]

$$R_1 \leq \frac{1}{2} \log \left(1 + \frac{P_1}{\sigma^2} \right), \tag{8.11}$$

$$R_2 \leq \frac{1}{2} \log \left(1 + \frac{P_2}{\sigma^2} \right), \tag{8.12}$$

$$R_1 + R_2 \leq \frac{1}{2} \log \left(1 + \frac{P_1 + P_2}{\sigma^2} \right). \tag{8.13}$$

Any rate pair (R_1, R_2) within the capacity region can be achieved using Shannon-style random coding, with independent codebooks for each of the two users. Here the entries of the two codebooks are generated $\sim_{i.i.d.} \mathcal{N}(0, P_1)$ and $\sim_{i.i.d.} \mathcal{N}(0, P_2)$, respectively. Each user transmits a codeword from their own codebook, and the receiver attempts to recover the two codewords from $y = x^{(1)} + x^{(2)} + w$ via *joint* maximum-likelihood decoding.

We now show how an efficient SPARC coding scheme can be used to communicate reliably at any pair of rates within the capacity region.

8.6 SPARCs for the Gaussian multiple-access channel

Consider a rate pair (R_1, R_2) within the MAC capacity region (8.11)–(8.13). The SPARC construction is similar to that of the broadcast channel. The two user's codebooks are defined via SPARC design matrices A_1 and A_2 with parameters (n, M, L_1) and (n, M, L_2), respectively. The parameters are chosen such that

$$nR_1 = L_1 \log M, \qquad nR_2 = L_2 \log M. \tag{8.14}$$

Power allocation   The non-zero coefficient in section $\ell \in [L_1]$ of A_1 is $\sqrt{nP_{1\ell}}$, while that in section $\ell \in [L_2]$ of A_2 is $\sqrt{nP_{2\ell}}$. With optimal (ML) decoding, each transmitter can use a flat power allocation across sections, i.e., $P_{1\ell} = P_1/L$ and $P_{2\ell} = P_2/L$ for $\ell \in [L]$.

For AMP decoding, we design a power allocation for the combined SPARC defined by the design matrix $A = [A_1\, A_2]$, and then partition the allocation between the two transmitters. Designing a combined allocation facilitates effective joint decoding of both the messages by the receiver. The details of designing such an allocation are given in the next section.

Encoding   Each transmitter $i \in \{1, 2\}$ first maps its message to a message vector $\beta^{(i)} \in \mathcal{B}_{M,L_i}$, and then generates its codeword $x^{(i)} = A_i \beta^{(i)}$. The channel output sequence at the receiver is

$$y = A_1 x^{(1)} + A_2 x^{(2)} + w = [A_1 \quad A_2] \begin{bmatrix} \beta^{(1)} \\ \beta^{(2)} \end{bmatrix} + w. \tag{8.15}$$

Optimal decoding   The receiver jointly decodes the two message vectors as

$$[\hat{\beta}_{\text{opt}}^{(1)}, \hat{\beta}_{\text{opt}}^{(2)}] = \underset{\hat{\beta}^{(1)} \in \mathcal{B}_{M,L_1}, \hat{\beta}^{(2)} \in \mathcal{B}_{M,L_2}}{\arg \min} \|y^{(2)} - A_1 \hat{\beta}^{(1)} - A_2 \hat{\beta}^{(2)}\|^2. \tag{8.16}$$

Writing $L = L_1 + L_2$, $A = [A_1\, A_2]$, and $\beta = \begin{bmatrix} \beta^{(1)} \\ \beta^{(2)} \end{bmatrix}$, an equivalent representation of the optimal decoder is

$$\hat{\beta}_{\text{opt}} = \underset{\hat{\beta} \in \mathcal{B}_{M,L}}{\arg \min} \|y - A\hat{\beta}\|^2. \tag{8.17}$$

The first L_1 sections of $\hat{\beta}_{\text{opt}}$ form $\hat{\beta}_{\text{opt}}^{(1)}$, while the remaining sections form $\hat{\beta}_{\text{opt}}^{(2)}$.

AMP decoding   The receiver runs a standard SPARC AMP decoding routine (as described in Section 3.4) on the concatenated design matrix $A = [A_1\, A_2]$. The first L_1 sections of the decoded message vector constitute $\hat{\beta}^{(1)}$, and the next L_2 constitute $\hat{\beta}^{(2)}$.

8.7 Power allocation for AMP decoding

As there is a single combined decoder, we first construct an overall power allocation with total power $P = P_1 + P_2$ for the concatenated SPARC $A = [A_1\ A_2]$, which has $L = L_1 + L_2$ sections and rate $R = R_1 + R_2$.
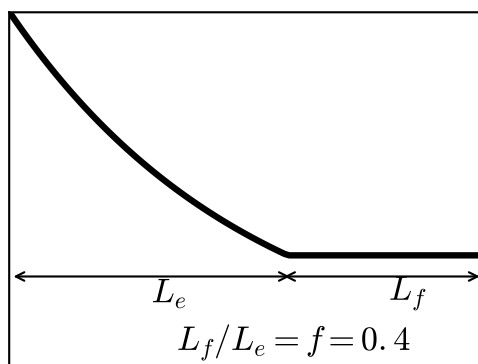
This overall power allocation $\{P_\ell\}_{\ell \in [L]}$ is constructed using the iterative technique described in Chapter 4 for point-to-point channels. Now, this power allocation must now partitioned between the two transmitters. We will use the fact that $\{P_\ell\}_{\ell \in [L]}$ in non-increasing in ℓ.

Transmitter $i \in \{1, 2\}$ must be allocated precisely L_i sections, whose total power must be no more than P_i. Additionally, we would like for any section errors to be fairly distributed between transmitters, so that each transmitter experiences approximately the same error rate. We know from Chapter 4 that the majority of errors occur in sections towards the end of the power allocation, where the power per section is lower and many sections share the same power (the *flat* region). Therefore we would like each user to have the same proportion of their allocation be flat. To summarize, the requirements for the power allocation are:
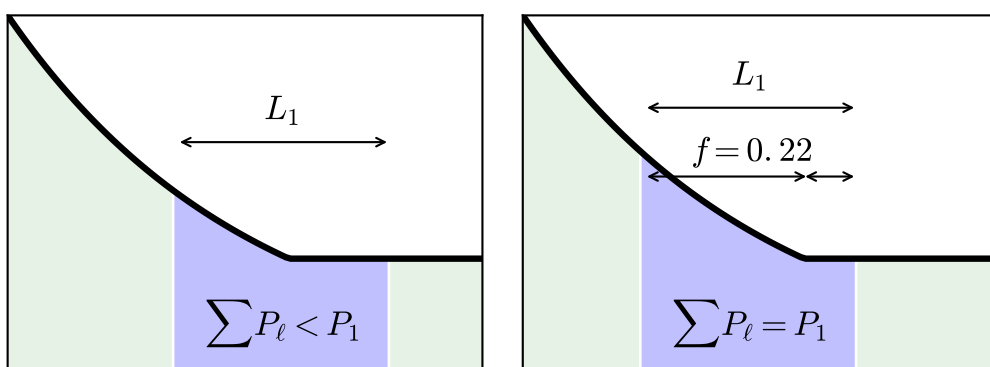
1. Of the $L = L_1 + L_2$ sections, we must allocate any L_1 sections to user 1, and the remaining L_2 to user 2.

2. The sections allocated to user 1 must sum to no more than P_1, and those for user 2 to no more than P_2.

3. Once the conditions above are met, choose the solution which divides any equal power sections more equally between the two users.

The strategy for partitioning the power allocation is as follows. We locate a bracket of size either L_1 or L_2 sections inside $\{P_\ell\}_{\ell \in [L]}$ such that its sum is as close as possible to, without exceeding, P_1 or P_2 respectively, and allocate the coefficients within the bracket to transmitter 1 or transmitter 2 as appropriate. The remaining sections on either side of the bracket are allocated to the other transmitter. The choice of bracket size (and thus of which transmitter is allocated the bracketed coefficients) is determined by which option gives the closest to optimal division of the coefficients from the flat section. This strategy is illustrated graphically in Figure 8.3.
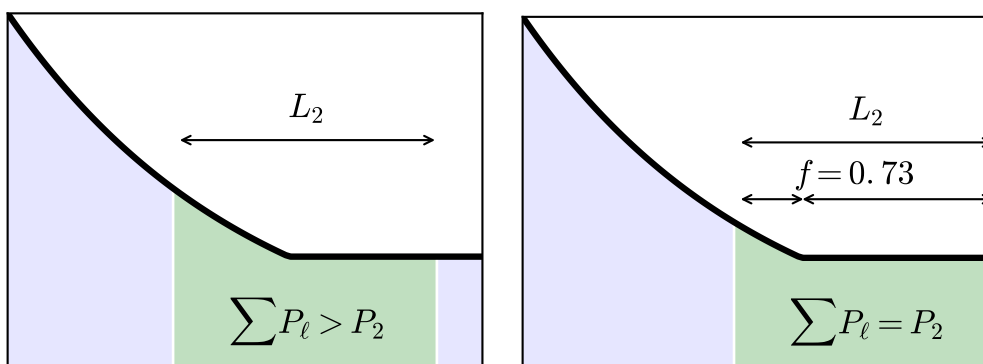
Remark 8.1. *The partitioning method above can be applied to any overall power allocation $\{P_\ell\}_{\ell \in [L]}$, including the exponentially decaying one where $P_\ell \propto e^{-2\mathcal{C}\ell/L}$ for $\ell \in [L]$. (Here $\mathcal{C}$ is the sum capacity $\frac{1}{2}\log(1 + P/\sigma^2)$, where $P = P_1 + P_2$.) The decoding analysis is identical to that of a point-to-point AWGN channel with power constraint P operating at rate $R = R_1 + R_2$. Therefore the AMP decoding result of Theorem 3.3 can be directly applied, and (3.72) bounds the probability of the sum of the section error rates of the two users exceeding some $\epsilon > 0$. This establishes that all rate pairs within the capacity region are achievable with an efficient AMP decoder.*

(a) The overall power allocation to partition.The ratio of exponential to flat sections is denoted f, here with $f = 0.4$.

(b) We first consider a bracket of size L_1, shown in blue. Our first attempt (on the left) contains too little power, so we move it leftwards until the contained power reaches P_1, shown on the right. At this position, $f = 0.22$.

(c) Next we consider a bracket of size L_2, shown in green. The first attempt contains too much power, so we move it rightwards until the contained power is P_2. In this position $f = 0.73$. Since the L_1 bracket has an f closer to that of the original allocation, we use that bracket for partitioning.

Figure 8.3: Example of the power allocation partitioning strategy. Sections highlighted in blue are considered for user 1, and those in green for user 2.
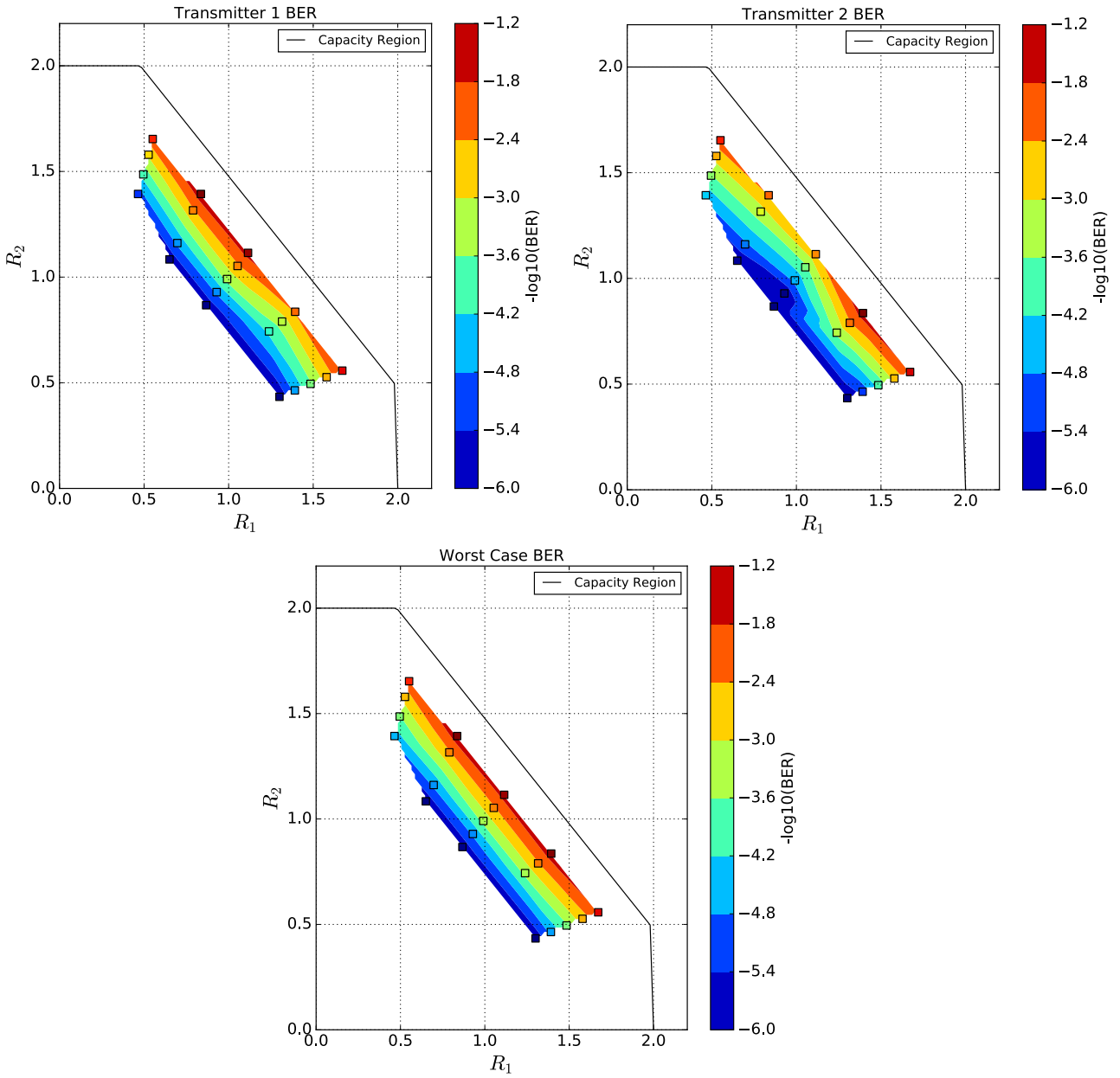
Figure 8.4: Bit error rates for multiple access channel simulations. Each displayed point is the average of approximately 30000 trials. $P_1 = 15$, $P_2 = 15$, $\sigma^2 = 1$, $M = 512$, $n = 4095$.

8.8 Simulation results

We now discuss the empirical performance of SPARCs with AMP decoding for the Gaussian MAC, considering a symmetric setup with $P_1 = P_2 = 15$, and $\sigma^2 = 1$. The sum-capacity is $\mathcal{C} = \frac{1}{2}\log\left(1 + \frac{P_1+P_2}{\sigma^2}\right)$ Each operating point is set as follows. We fix $\gamma \in [0,1]$ and then set $R_1 + R_2 = R = \gamma\mathcal{C}$. The parameter γ represents the backoff from the sum rate limit. Next fix $\alpha \in [0,1]$ which sets the share of this sum rate for each transmitter as $R_1 = \alpha R$ and $R_2 = (1-\alpha)R$.

For the SPARC design matrix, first fix $M = 512$ and $n = 4095$. The parameters L_1, L_2 are then determined by the rate pair: $L_1 = nR_1/\log M$, and $L_2 = nR_2/\log M$. A power allocation is then designed using the sum rate R and $L = L_1 + L_2$, and partitioned according to the strategy described in the previous section.

Figure 8.4 shows the bit error rates with AMP decoding for different values of (R_1, R_2). The boundary of the capacity region is also shown. After performing AMP decoding on the combined SPARC A, the number of sections decoded in error is counted separately for the first L_1 and then the last L_2 sections, and used to report section error rates for each transmitter. We also report the worst case section error rate between the two users.

As the experimental set-up is equivalent to a single point-to-point channel with the same sum power and sum rate and power allocation, we expect to obtain similar bit error performance to that scenario. The results support this, with good bit error rates even reasonably close to capacity at all points in the rate region. The worst case bit error rate is also close to each individual user's bit error rate, showing that the power allocation partition is generally successful at ensuring equal error rates between users.
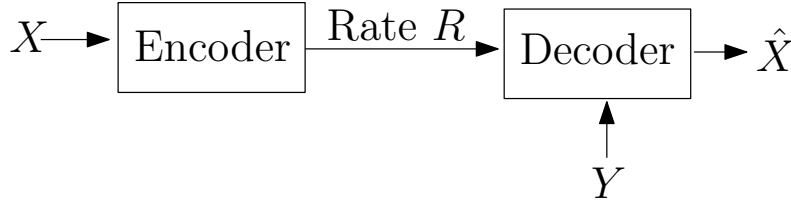
Chapter 9

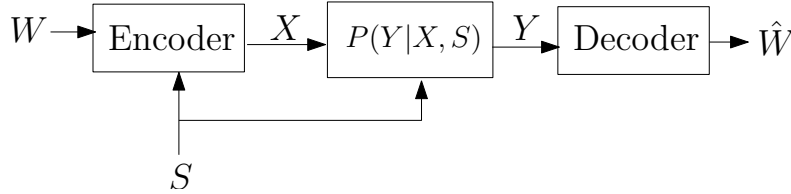# Communication and Compression with Side Information

Random binning is a key ingredient of optimal coding schemes for several models in multiuser information theory. In a scheme with binning, a large codebook is partitioned into equal-sized codebooks of smaller rate. In this chapter, we demonstrate how random binning can be efficiently implemented using SPARCs for two canonical multiuser models: lossy compression with side information at the decoder (the Wyner-Ziv model [117]), and channel coding with state information at the encoder (the Gelfand-Pinsker model [47, 29]). We will consider the Gaussian versions of these models, and aim to construct SPARC-based coding schemes that achieve rates close to the information-theoretic optimum. The proposed binning construction can be extended to other multiuser models such as lossy distributed source coding where the best known achievable rates use a random binning strategy.

In the Wyner-Ziv compression problem (Figure 9.1a), given a rate $R > 0$, the goal is to design a coding scheme that optimally uses the decoder side information Y to reconstruct the source with minimal distortion. This model has been used in distributed video coding [48, 90]. In the Gelfand-Pinsker channel coding problem (Figure 9.1b), the encoder knows the channel state S non-causally (at the beginning of communication), while the decoder only receives the channel output Y. The goal is to design a scheme that optimally uses the channel state information available at the encoder to achieve the maximal rate. The Gelfand-Pinsker problem is the channel coding dual of the Wyner-Ziv problem [88, 18], and has been used in multi-antenna communication [103], digital watermarking [84, 25] and steganography [86].

We will consider the Gaussian versions of the Wyner-Ziv and the Gelfand-Pinsker models, where the side information and the additive noise are independent Gaussian random variables. An interesting feature of the Gaussian Wyner-Ziv and Gelfand-Pinsker problems is that the optimal rate in each case is the same as the setting where the side/state information is available to both the encoder and the decoder [117, 29].

(a) Compressing X with decoder side-information Y.

(b) Communicating a message W over channel $P(Y|XS)$ with state S known at the encoder.

Previous code constructions Several practical coding schemes have been proposed for the Wyner-Ziv problem, e.g., [89, 92, 119]. Recent constructions based on polar codes are the first computationally efficient schemes that are provably rate-optimal [69, 70]. However, the polar coding constructions are only applicable to problems where the source and side-information distributions are discrete and symmetric. For the Wyner-Ziv problem with continuous source and side-information distributions, elegant coding schemes such as those based on lattices [122, 39] have been proposed. But these generally have exponential encoding and decoding complexity. Constructions using nested lattice codes have also been used for the Gaussian Gelfand-Pinsker model (dubbed 'writing on dirty paper') [122, 40, 25]. Computationally efficient code designs for this problem have been proposed in several works, e.g., [104, 41].

9.1   Binning with SPARCs

We now describe the SPARC binning construction, which is applied to the Wyner-Ziv and Gelfand-Pinsker problems in the next two sections. For any pair of rates R_1, R with $R_1 > R$, we would like to divide a rate R_1 SPARC with block length n into e^{nR} bins. Each bin corresponds to a rate $(R_1 - R)$ SPARC with the same block length n. This is done as follows.

Fix the parameters M, L, n of the design matrix A such that

$$L \log M = nR_1. \tag{9.1}$$

As shown in Figure 9.2, divide each section of A into sub-sections consisting of M' columns each. Each bin is a smaller SPARC defined via a *sub-matrix* of A formed by picking one sub-section from each section. For example, the collection of shaded sub-sections in the figure together forms one bin. If M' is chosen such that
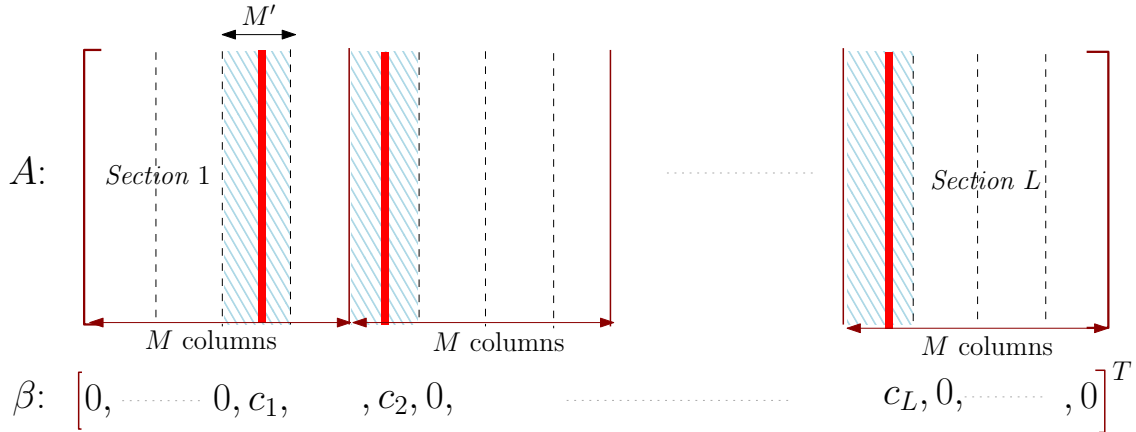
$$n(R_1 - R) = L \log M', \tag{9.2}$$

Figure 9.2: Codebook binning using SPARCs

then each sub-matrix defines a bin that is a rate $(R_1 - R)$ SPARC with parameters (n, L, M'). Since we have (M/M') sub-section choices in each of the L sections, the total number of bins that can be chosen is $(M/M')^L$. Combining (9.1) and (9.2), we have

$$L \log \frac{M}{M'} = nR, \quad \text{or} \quad \left(\frac{M}{M'} \right)^L = e^{nR} \tag{9.3}$$

Therefore, the number of bins is e^{nR}, as required. The binning structure mimics that of the SPARC codebook, where two codewords may share up to $(L-1)$ columns. Similarly, two bins may share as many as $(L-1)$ sub-sections of A.

9.2 Wyner-Ziv coding with SPARCs

Consider the model in Figure 9.1a, with $X \sim \mathcal{N}(0, \sigma^2)$ being an i.i.d. Gaussian source to be compressed with mean-squared distortion D. The decoder side-information Y is noisy version of X and is related to X by $Y = X + Z$, where $Z \sim \mathcal{N}(0, N)$ is independent of X. Let the target distortion be $D < \text{Var}(X|Y)$. (Distortions greater than this value can be achieved with zero rate, by simply estimating X from Y.)

Let $x, y \in \mathbb{R}^n$ be the source and side information sequences drawn i.i.d. according to the distributions of X and Y, respectively. The sequence y is available at the decoder non-causally. If y were available at the encoder as well, then an optimal strategy is to compress $(y - x)$ with a rate high enough to ensure reconstruction of x within distortion D. The minimum rate required for this strategy is $\frac{1}{2} \log \frac{\text{Var}(X|Y)}{D}$ nats/sample. The result of Wyner and Ziv [117] shows that this rate is achievable even when y is available only at the decoder.

We first review the main ideas in the Wyner-Ziv random coding scheme. Define an auxiliary random variable U jointly distributed with X according to

$$U = X + V, \tag{9.4}$$

137

where $V \sim \mathcal{N}(0, Q)$ is independent of X. We choose

$$Q = \left(\frac{1}{D} - \frac{1}{\text{Var}(X|Y)} \right)^{-1}. \tag{9.5}$$

One can invert this test channel to write

$$X = \frac{\sigma^2}{\sigma^2 + Q} U + V' = U' + V', \tag{9.6}$$

where $V' \sim \mathcal{N}(0, \frac{\sigma^2 Q}{\sigma^2 + Q})$ is independent of $U' \sim \mathcal{N}(0, \frac{\sigma^4}{\sigma^2 + Q})$.

In the Wyner-Ziv scheme [117], x is first quantized to a codeword u' within distortion $\frac{\sigma^2 Q}{\sigma^2 + Q}$, using a rate-distortion codebook of rate R_1. The sequences in this codebook, say $\{u'(1), \ldots, u'(e^{nR_1})\}$, are drawn $\sim_{i.i.d.} \mathcal{N}(0, \frac{\sigma^4}{\sigma^2 + Q})$. Shannon's rate-distortion theorem guarantees that with high probability a codeword will be found within the required distortion if

$$R_1 > I(X; U') = I(X; U) = \frac{1}{2} \log \left(1 + \frac{\sigma^2}{Q} \right), \tag{9.7}$$

where the mutual information has been calculated using the test channel in (9.6). The index corresponding to the chosen codeword u' is not sent to the decoder directly. Instead, the size e^{nR_1} rate-distortion codebook is divided into e^{nR} equal-sized bins, with a uniformly random assignment of sequences to bins. The encoder only sends the index of the bin containing u', which requires a rate of R nats/sample .

The decoder's task is to recover u' using the bin index and the side information y. This is equivalent to a *channel decoding* problem, where the effective channel is

$$Y = X + Z = U' + V' + Z. \tag{9.8}$$

The snr of the effective channel is

$$\frac{\text{Var}(U')}{\text{Var}(V') + \text{Var}(Z)} = \frac{\sigma^4}{\sigma^2 Q + (\sigma^2 + Q) N}. \tag{9.9}$$

The decoder will succeed with high probability if the number of sequences within the bin, $e^{n(R_1 - R)}$, is less than $e^{nI(U'; Y)}$. We therefore need

$$R_1 - R < I(U'; Y) = \frac{1}{2} \log \left(1 + \frac{\sigma^4}{\sigma^2 Q + (\sigma^2 + Q) N} \right). \tag{9.10}$$

Combining the conditions (9.7) and (9.10), we conclude that both the encoding and the decoding steps are successful with high probability if

$$R > I(X; U') - I(Y; U') = \frac{1}{2} \log \frac{\text{Var}(X|Y)}{D}, \tag{9.11}$$

where the last equality follows by substituting the value for Q from (9.5). For any R satisfying (9.11), $R_1 > R$ can be chosen to satisfy both (9.7) and (9.10).

The final step at the decoder is produce the reconstructed sequence $\hat{x}$ as the MMSE estimate of x given u' and y:

$$\hat{x} = \left(\frac{1}{Q} + \frac{1}{\sigma^2} + \frac{1}{N}\right)^{-1} \left(\frac{u'}{Q} + \frac{y}{N}\right). \tag{9.12}$$

Since the sequence triple (x, y, u') is jointly typical according to the joint distribution of the random variables (X, Y, U'), the expected distortion $\mathbb{E}|x - \hat{x}|^2$ can be made arbitrarily close to D for sufficiently large n.

We now implement the coding scheme using a SPARC with optimal (least-squares) encoding and decoding. For a given target distortion D, choose $R > \frac{1}{2}\log\frac{\text{Var}(X|Y)}{D}$, and choose R_1 such that

$$\frac{1}{2}\log\left(1 + \frac{\sigma^2}{Q}\right) < R_1 < R + \frac{1}{2}\log\left(1 + \frac{\sigma^4}{\sigma^2 Q + (\sigma^2 + Q)N}\right), \tag{9.13}$$

so that both (9.7) and (9.10) are satisfied.

Fix block length n, and consider a SPARC defined via an $n \times ML$ design matrix A with $M = L^b$ and $bL\log L = nR_1$, where b is greater than the minimum value specified by Theorem 6.2. The entries of A are $\sim_{i.i.d.} \mathcal{N}(0, \frac{1}{n})$, and the non-zero entries each $\beta \in \mathcal{B}_{M,L}$ are all set to $\sqrt{\frac{n}{L}\frac{\sigma^4}{(\sigma^2 + Q)}}$.

With M' be determined by $M'^L = e^{n(R_1 - R)}$, partition each section of A into sub-sections of M' columns each, as shown in Figure 9.2.

Encoding: The encoder determines the SPARC codeword that is closest in Euclidean distance to the source sequence x. Let

$$\beta^* = \underset{\beta \in \mathcal{B}_{M,L}}{\arg\min} \|x - A\beta\|^2. \tag{9.14}$$

The encoder sends the decoder a message $W \equiv (p_1, \ldots, p_L)$, where $p_i \in \{1, \ldots, \frac{M}{M'}\}$ indicates the subsection in the ith section of A where β^* contains a non-zero element.

Decoding: Let $A_{\text{bin}(W)}$ be the $n \times M'L$ sub-matrix corresponding to the subsections of A corresponding to the bin index W. Recalling from Section 9.1 that $A_{\text{bin}(W)}$ defines an $n \times M'L$ SPARC design matrix, the decoder determines

$$\hat{\beta} = \underset{\beta \in \mathcal{B}_{M',L}}{\arg\min} \|y - A_{\text{bin}(W)}\beta\|^2. \tag{9.15}$$

Letting $\hat{u} = A_{\text{bin}(W)}\hat{\beta}$, the source sequence is reconstructed as

$$\hat{x} = \left(\frac{1}{Q} + \frac{1}{\sigma^2} + \frac{1}{N}\right)^{-1} \left(\frac{\hat{u}}{Q} + \frac{y}{N}\right). \tag{9.16}$$

Analysis: The SPARC rate-distortion result in Theorem 6.2 guarantees that the encoding will succeed with high probability. That is, for R_1 exceeding the lower bound in (9.13), the probability of the event

$$\left\{\frac{1}{n}\|x - A\beta^*\|^2 > \frac{\sigma^2 Q}{\sigma^2 + Q}\right\}$$

139

is exponentially small in n. On the decoder side, the effective channel is given by (9.8), with the task being to recover the codeword u' with $v' + z$ treated as a noise sequence. If we assume that $v' + z$ is distributed as $\sim_{i.i.d.} \mathcal{N}(0, \text{Var}(V') + \sigma^2)$, then the SPARC channel coding result Theorem 2.1 guarantees reliable decoding with exponentially small error probability. However, this assumption is not true at finite code lengths. Indeed, recall that $v = (x - u')$ is the distortion (quantization noise) incurred at the encoder. Therefore, v' may be dependent on the codeword u'; moreover, its distribution will not be exactly Gaussian.

Though we expect that the distribution of $(v' + z)$ will be asymptotically independent of u' and converge to Gaussian, a careful analysis is needed to rigorously prove that SPARCs achieve the Gaussian Wyner-Ziv rate-distortion bound. This remains an open question for future work.

9.3   Gelfand-Pinsker coding with SPARCs

Consider the model in Figure 9.1b, with the channel law $P(Y|X, S)$ given by

$$Y = X + S + Z. \tag{9.17}$$

The state variable $S \sim \mathcal{N}(0, \sigma_s^2)$ is independent of the additive noise $Z \sim \mathcal{N}(0, N)$. There is an average power constraint P on the input sequence $x \in \mathbb{R}^n$. The state sequence $s \in \mathbb{R}^n$ is $\sim_{i.i.d.} \mathcal{N}(0, \sigma_s^2)$, and is known non-causally at the encoder.

Due to the power constraint, the encoder cannot simply cancel out the effect of the state sequence s using the codeword. In Costa's capacity-achieving scheme [29], one part of the state sequence s is used by the encoder to produce the input sequence x, and the remaining part is treated as noise. This is done as follows.

Define an auxiliary random variable U as

$$U = X + \alpha S, \tag{9.18}$$

where $X \sim \mathcal{N}(0, P)$ is independent of $S \sim \mathcal{N}(0, \sigma_s^2)$, and

$$\alpha = \frac{P}{P + N}. \tag{9.19}$$

Inverting the test channel U, we write

$$S = \frac{\alpha \sigma_s^2}{P + \alpha^2 \sigma_s^2} U + X' = U' + X' \tag{9.20}$$

where $U' \sim \mathcal{N}(0, \frac{\alpha^2 \sigma_s^4}{P + \alpha^2 \sigma_s^2})$ and $X' \sim \mathcal{N}(0, \frac{P \sigma_s^2}{P + \alpha^2 \sigma_s^2})$ are independent.

In Costa's scheme [29], we first construct a random codebook of rate R_1, with sequences $\{u'(1), \ldots, u'(e^{nR_1})\}$ drawn $\sim_{i.i.d.} \mathcal{N}(0, \frac{\alpha^2 \sigma_s^4}{P + \alpha^2 \sigma_s^2})$. The codebook is partitioned into e^{nR} bins, with each bin containing $e^{n(R_1 - R)}$ codewords.

To transmit message $W \in \{1, \ldots, e^{nR}\}$, the encoder finds a codeword u' *inside bin* W of the codebook that is within distortion $\frac{P\sigma_s^2}{P+\alpha^2\sigma_s^2}$ of the state sequence s. From Shannon's rate-distortion theorem, such a codeword will be found with high probability if

$$R_1 - R > I(S; U') = \frac{1}{2} \log \left(1 + \frac{\alpha^2 \sigma_s^2}{P} \right). \tag{9.21}$$

The transmitted sequence x is then determined as

$$x = \left(\frac{P + \alpha^2 \sigma_s^2}{\alpha \sigma_s^2} \right) u' - \alpha s. \tag{9.22}$$

Since the empirical joint distribution of (u', s) is close to that specified by the test channel (9.20), it can be verified that the second moment of x will be close to P with high probability.

Using the test channel in (9.20), the channel input-output relationship can be expressed as

$$\begin{aligned} Y &= X + S + Z \\ &= \left(\frac{P + \alpha^2 \sigma_s^2}{\alpha \sigma_s^2} \right) U' + (1 - \alpha) S + Z \\ &= \left(\frac{P + \alpha^2 \sigma_s^2}{\alpha \sigma_s^2} + 1 - \alpha \right) U' + (1 - \alpha) X' + Z, \end{aligned} \tag{9.23}$$

where U', X', Z' are mutually independent.

The decoder's task is to determine the codeword u' from the output sequence y. The index of the bin containing the decoded codeword gives the reconstructed message $\hat{W}$. Assuming that the encoding has been successful, the empirical joint distribution of (u', y) will be close to that of the effective channel in (9.23). The effective signal to noise ratio of this channel is

$$\mathsf{snr}_{\text{eff}} = \frac{\frac{(P+\alpha\sigma_s^2)^2}{\alpha^2\sigma_s^4} \text{Var}(U')}{(1-\alpha)^2 \text{Var}(X') + \text{Var}(Z)} = \frac{(P + \alpha\sigma_s^2)^2}{(1-\alpha)^2 P\sigma_s^2 + N(P + \alpha^2\sigma_s^2)}. \tag{9.24}$$

The decoding step will be successful with high probability if

$$\begin{aligned} R_1 < I(U'; Y) &= \frac{1}{2} \log(1 + \mathsf{snr}_{\text{eff}}) \\ &= \frac{1}{2} \log \left(\frac{(P + \alpha^2 \sigma_s^2)(\sigma_s^2 + P + N)}{(1 - \alpha)^2 P \sigma_s^2 + N(P + \alpha^2 \sigma_s^2)} \right). \end{aligned} \tag{9.25}$$

Combining (9.21) and (9.25), we conclude that both the encoding and the decoding steps are successful with high probability if

$$R < I(S; U') - I(U'; Y) = \frac{1}{2} \log \left(1 + \frac{P}{N} \right), \tag{9.26}$$

where the last equality follows by substituting $\alpha = \frac{P}{P+N}$ and simplifying. For any R satisfying (9.26), R_1 should be chosen large enough to satisfy (9.25).

We now implement the above coding scheme using a SPARC with optimal (minimum distance) encoding and decoding. Fix block length n, and consider a SPARC defined via an $n \times ML$ design matrix A with $M = L^b$ and $bL \log L = nR_1$, where b is greater than the minimum value specified by Theorem 6.2. The entries of A are $\sim_{i.i.d.} \mathcal{N}(0, \frac{1}{n})$, and the non-zero entries each $\beta \in \mathcal{B}_{M,L}$ are all set to $\sqrt{\frac{n}{L} \frac{\alpha^2 \sigma_s^4}{(P + \alpha^2 \sigma_s^2)}}$.

With M' be determined by $M'^L = e^{n(R_1 - R)}$, partition each section of A into sub-sections of M' columns each, as shown in Figure 9.2. This defines e^{nR} bins, one for each message.

Encoding: To transmit message $W \in [e^{nR}]$, the encoder determines the SPARC codeword within bin W that is closest to the state sequence s. Denoting by $A_{\mathsf{bin}(W)}$ the sub-matrix of A corresponding to bin W, the encoder computes

$$\beta^* = \arg\min_{\beta \in \mathcal{B}_{M',L}} \|s - A_{\mathsf{bin}(W)}\beta\|^2.$$

Following (9.22), the transmitted sequence is

$$x = \left(\frac{P + \alpha^2 \sigma_s^2}{\alpha \sigma_s^2} \right) A_{\mathsf{bin}(W)} \beta^* - \alpha s.$$

Decoding: The decoder determines the codeword in the big rate R_1 SPARC closest to the output sequence y. It computes

$$\hat{\beta} = \arg\min_{\beta \in \mathcal{B}_{M,L}} \|y - A\beta\|^2. \tag{9.27}$$

The index of the bin containing $\hat{\beta}$ is the decoded message $\hat{W}$.

Analysis: The SPARC rate-distortion result in Theorem 6.2 guarantees that encoding will succeed with high probability provided $R_1 - R$ satisfies (9.21). However the analysis of the decoder is challenging, for reasons similar to those for Wyner-Ziv SPARCs. In the effective channel (9.23), we cannot assume that the noise $(1 - \alpha)x' + z$ is exactly i.i.d. Gaussian and independent of the codeword u'. This is because $x' = s - u'$, the quantization noise incurred at the encoder, cannot be assumed to be independent of u' and Gaussian.

In summary, a rigorous analysis of the SPARC coding schemes for the Gaussian Gelfand-Pinsker and Wyner-Ziv models remains open. Another important open problem is to construct feasible SPARC coding schemes that achieve the Shannon limits for these models. The challenges in designing such schemes, and some ideas to address them, are discussed in the final chapter.

Chapter 10

Open Problems and Further Directions

In the first nine chapters, we described how sparse regression codes can be used for channel coding, lossy source coding, and multi-terminal versions of these problems. We conclude the monograph with a discussion of open questions and directions for further work.

10.1   Channel coding with SPARCs

Gap from capacity   For fixed value of decoding error probability, how fast can the gap from capacity $(\mathcal{C} - R)$ shrink with growing block length n? With optimal decoding, the result in Chapter 2 implies that the gap from capacity for SPARCs is of $O(\frac{1}{\sqrt{n}})$, which is order-optimal (from the results in [87]). In contrast, the feasible decoders in Chapter 3 all have much larger gap from capacity: even with optimized power allocations, the gap to capacity is no smaller than $O(\frac{\log \log n}{\log n})$.

A key open question is: can one achieve polynomial gap to capacity using SPARCs with feasible decoding? For polar codes over binary input symmetric channels, This question was recently answered in the affirmative [55]. Achieving a polynomial gap from capacity for SPARCs may require new constructions such as spatially coupled SPARCs with power allocation as well as new decoding algorithms.

Analysis of sub-Gaussian and Hadamard-based SPARCs   In Chapter 2, we analyzed the optimal decoder for SPARCs defined via i.i.d. Gaussian and Bernoulli dictionaries. An interesting direction is to generalize these results to dictionaries with arbitrary i.i.d. sub-Gaussian random variables. The key part of the analysis in the Gaussian case involves controlling the moment generating function of the difference between the squares of two Gaussian random variables. Extending the analysis to sub-Gaussian dictionaries would involve reworking the proof of Proposition 2.1 to replace the steps using Gaussian-specific results with the appropriate results for sub-Gaussians.

A more difficult open question is to analyze SPARC dictionaries defined via partial Hadamard or Fourier matrices. As described in Chapter 4, these structured matrices significantly reduce decoding

and storage complexity, and are therefore important for practical implementation of SPARCs.

The analysis of feasible SPARC decoders with non-Gaussian dictionaries is more challenging than that of optimal decoding, and remains open even for Bernoulli dictionaries.

Coded modulation using SPARCs The empirical results in Section 4.5 show that concatenating an outer LPDC code with an inner SPARC can produce a steep drop in error rates, when the snr exceeds a threshold. This waterfall behaviour was obtained using an off-the-shelf LDPC code. It appears likely that one can further improve the error performance (i.e., obtain a smaller threshold) by jointly optimizing the design of the outer LDPC code and inner SPARC using an EXIT chart analysis, similar to [108].

SPARCs for general channel models The recent work of Barbier et al. [9, 10] extends the spatially coupled SPARC construction to general memoryless channels. The idea is to apply a symbol-by-symbol mapping to each Gaussian SC-SPARC codeword $A\beta$ to produce a codebook whose input alphabet and distribution are tailored to the channel. The message vector β is recovered from the channel output sequence using generalized AMP (GAMP) [91] decoding. The fixed points of the state evolution recursion are analyzed using the potential function method in [10]. This analysis predicts that the GAMP decoder is asymptotically capacity achieving for a general memoryless channel.

The results in [10] suggest two directions for further work. One is to extend the analysis of spatially coupled SPARCs to general memoryless channels with GAMP decoding, and rigorously prove that the coding scheme is capacity achieving. It would also be interesting to study the empirical performance of the SPARC coding scheme over commonly studied memoryless channels such as the binary symmetric channel, and compare the performance with that of capacity achieving codes such as LDPC and polar codes.

Another interesting direction is to generalize SPARC coding schemes originally designed for AWGN channels to Gaussian fading channels and MIMO channels, which are important in wireless communication [111, 50].

10.2   Lossy compression with SPARCs

Gap from optimal rate-distortion function With optimal encoding, the results in Chapter 6 imply that for a given distortion level D, the SPARC rate R should be $O(\frac{1}{\sqrt{n}})$ higher than $R^*(D)$ (the optimal Gaussian rate-distortion function) in order to ensure a fixed probability of excess distortion. In contrast, the successive cancellation encoder described in Chapter 7 can only achieve rates that are $O(\frac{\log \log n}{\log n})$ above $R^*(D)$ (see Sec. 7.3.1).

A key open problem is to design a feasible SPARC encoder with a gap from $R^*(D)$ that is of smaller order. One idea is to investigate algorithms that process multiple sections at a time and use soft-decision updating, instead of encoding one section at a time. Another idea to improve the

gap from $R^*(D)$ is to use spatially coupled SPARCs for lossy compression. Doing so would also yield coding schemes with spatially coupled SPARCs for multiuser models that require random binning.

Sub-Gaussian and structured dictionaries   The compression performance of sub-Gaussian or Hadamard-based SPARC design matrices is empirically very similar to that of i.i.d. Gaussian design matrices. As in the channel coding case, Hadamard-based designs significantly reduce both the encoding complexity and the memory required. However, there are no theoretical results for compression with non-Gaussian dictionaries. It would of interest to establish performance guarantees for compression with such dictionaries, both with optimal encoding and with successive cancellation encoding.

Lossy compression of general sources   We expect that the results for SPARC compression of i.i.d. Gaussian sources in Chapters 6 and 7 can be extended to Gaussian sources with memory (e.g., Gauss-Markov sources), using the spectral representation of the source distribution. More broadly, can one use SPARC-like constructions to compare finite alphabet sources, e.g., binary sources with Hamming distortion?

10.3   Multi-terminal coding schemes with SPARCs

Performance guarantees with optimal encoding   A key open problem is to provide a rigorous proof that the SPARC coding schemes in Chapter 9 for the Wyner-Ziv and Gelfand-Pinsker problems achieve the Shannon limits. As discussed in Sections 9.2 and 9.3, the main challenge in the analysis of both schemes is that part of the effective noise at the decoder is quantization noise, which cannot be assumed to be Gaussian and independent of the codeword.

Performance guarantees with feasible encoding   The coding schemes for the Wyner-Ziv and Gelfand-Pinsker problems each involve a quantization operation at the encoder and a channel decoding operation at the decoder. Both operations are to be performed using the same overall SPARC. The key challenge in constructing a rate-optimal feasible coding scheme based on power allocation is that the optimal allocations for the quantization and the channel decoding parts are different (as the rates for the two parts are different). Since the overall SPARC must have a single power allocation, we cannot simultaneously ensure that the power allocation is optimal for both the quantization and channel decoding operations. One idea to construct rate-optimal feasible coding schemes is to use spatially coupled SPARCs, which circumvent the need for power allocation. This will require first designing a lossy compression scheme using SC-SPARCs.

Coding schemes for general Gaussian multiuser models   The best-known rates for many canonical models in multiuser information theory are achieved by coding schemes that use superposition coding and/or random binning. Since we have shown how to implement both these operations using SPARCs, one can design SPARC-based schemes for a variety of Gaussian multi-terminal

problems such as the interference channel, distributed lossy compression, and multiple descriptions coding. In addition to the theoretical question of establishing rigorous performance guarantees for such schemes, an interesting direction for empirical work is to provide design guidelines to optimize the error performance at finite block lengths.

SPARC-based coding schemes have recently been used for unsourced random access communication over the AWGN channel [44]. Since SPARCs are built on the principle of superposition coding, we expect that they will be promising candidates for new variants of multiple-access or broadcast communication involving a large number of nodes.

Acknowledgements

References

[1] Python script for SPARC with AMP decoding.

[2] The coded modulation library, 2008.

[3] E. Abbe and A. Barron. Polar coding schemes for the AWGN channel. In *Proc. IEEE Int. Symp. Inf. Theory*, pages 194–198. IEEE, 2011.

[4] V. Aref, N. Macris, and M. Vuffray. Approaching the rate-distortion limit with spatial coupling, belief propagation, and decimation. *IEEE Trans. Inf. Theory*, 61(7):3954–3979, 2015.

[5] E. Arikan. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Trans. Inf. Theory*, 55(7):3051 –3073, July 2009.

[6] A. Ashikhmin, G. Kramer, and S. ten Brink. Extrinsic information transfer functions: model and erasure channel properties. *IEEE Trans. Inf. Theory*, 50(11):2657–2673, 2004.

[7] R. R. Bahadur and R. R. Rao. On deviations of the sample mean. *The Annals of Mathematical Statistics*, 31(4), 1960.

[8] J. Barbier, M. Dia, and N. Macris. Proof of threshold saturation for spatially coupled sparse superposition codes. In *Proc. IEEE Int. Symp. Inf. Theory*, 2016.

[9] J. Barbier, M. Dia, and N. Macris. Threshold saturation of spatially coupled sparse superposition codes for all memoryless channels. In *IEEE Inf. Theory Workshop*, 2016.

[10] J. Barbier, M. Dia, and N. Macris. Universal Sparse Superposition Codes with Spatial Coupling and GAMP Decoding. arXiv:1707.04203, July 2017.

[11] J. Barbier and F. Krzakala. Replica analysis and approximate message passing decoder for sparse superposition codes. In *Proc. IEEE Int. Symp. Inf. Theory*, 2014.

[12] J. Barbier and F. Krzakala. Approximate message passing decoder and capacity-achieving sparse superposition codes. *IEEE Trans. Inf. Theory*, 63(8):4894–4927, August 2017.

[13] J. Barbier, C. Schülke, and F. Krzakala. Approximate message-passing with spatially coupled structured operators, with applications to compressed sensing and sparse superposition codes. *Journal of Statistical Mechanics: Theory and Experiment*, (5), 2015.

[14] J. Barbier, C. Schülke, and F. Krzakala. Approximate message-passing with spatially coupled structured operators, with applications to compressed sensing and sparse superposition codes. *Journal of Statistical Mechanics: Theory and Experiment*, 2015(5):P05013, 2015.

[15] A. Barron and A. Joseph. Least squares superposition codes of moderate dictionary size, reliable at rates up to capacity. Arxiv:1712.06866, 2010.

[16] A. Barron and A. Joseph. Least squares superposition codes of moderate dictionary size are reliable at rates up to capacity. *IEEE Trans. Inf. Theory*, 58(5):2541–2557, Feb. 2012.

[17] A. R. Barron and S. Cho. High-rate sparse superposition codes with iteratively optimal estimates. In *Proc. IEEE Int. Symp. Inf. Theory*, 2012.

[18] R. Barron, B. Chen, and G. Wornell. The duality between information embedding and source coding with side information and some applications. *IEEE Trans. Inf. Theory*, 49(5):1159 – 1180, May 2003.

[19] M. Bayati and A. Montanari. The dynamics of message passing on dense graphs, with applications to compressed sensing. *IEEE Trans. Inf. Theory*, pages 764–785, 2011.

[20] M. Bayati and A. Montanari. The LASSO Risk for Gaussian Matrices. *IEEE Trans. Inf. Theory*, 58(4):1997–2017, April 2012.

[21] C. Berrou and A. Glavieux. Near optimum error correcting coding and decoding: turbo-codes. *IEEE Trans. Commun.*, 44(10):1261 –1271, Oct 1996.

[22] R. E. Blahut. *Algebraic codes for data transmission*. Cambridge University Press, 2003.

[23] G. Böcherer, F. Steiner, and P. Schulte. Bandwidth efficient and rate-matched low-density parity-check coded modulation. *IEEE Trans. Commun.*, 63(12):4651–4665, 2015.

[24] CCSDS. *131.0-B-2 TM Synchonization and Channel Coding*, August 2011.

[25] B. Chen and G. Wornell. Quantization index modulation: a class of provably good methods for digital watermarking and information embedding. *IEEE Trans. Inf. Theory*, 47(4):1423 –1443, May 2001.

[26] S. Cho. *High-dimensional regression with random design, including sparse superposition codes*. PhD thesis, Yale University, 2014.

[27] S. Cho and A. Barron. Approximate iterative bayes optimal estimates for high-rate sparse superposition codes. In *Sixth Workshop on Information-Theoretic Methods in Science and Engineering*, 2013.

[28] A. Coja-Oghlan and L. Zdeborová. The condensation transition in random hypergraph 2-coloring. In *Proc. 23rd Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 241–250, 2012.

[29] M. Costa. Writing on dirty paper (corresp.). *IEEE Trans. Inf. Theory*, 29(3):439 – 441, May 1983.

[30] D. J. Costello and G. D. Forney. Channel coding: The road to channel capacity. *Proc. IEEE*, 95(6):1150–1177, 2007.

[31] T. Cover. Broadcast channels. *IEEE Trans. Inf. Theory*, 18(1):2–14, 1972.

[32] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley and Sons, 2012.

[33] H. David and H. Nagaraja. *Order Statistics*. John Wiley & Sons, 2003.

[34] F. Den Hollander. *Large deviations*, volume 14. Amer. Mathematical Society, 2008.

[35] D. L. Donoho, A. Javanmard, and A. Montanari. Information-theoretically optimal compressed sensing via spatial coupling and approximate message passing. *IEEE Trans. Inf. Theory*, (11):7434–7464, Nov. 2013.

[36] D. L. Donoho, A. Maleki, and A. Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, 2009.

[37] A. El Gamal and Y.-H. Kim. *Network Information Theory*. Cambridge University Press, 2011.

[38] W. Equitz and T. Cover. Successive refinement of information. *IEEE Trans. Inf. Theory*, 37(2):269 –275, Mar 1991.

[39] U. Erez, S. Litsyn, and R. Zamir. Lattices which are good for (almost) everything. *IEEE Trans. Inf. Theory*, 51(10):3401–3416, 2005.

[40] U. Erez, S. Shamai, and R. Zamir. Capacity and lattice strategies for canceling known interference. *IEEE Trans. Inf. Theory*, 51(11):3820 – 3833, Nov. 2005.

[41] U. Erez and S. ten Brink. A close-to-capacity dirty paper coding scheme. *IEEE Trans. Inf. Theory*, 51(10):3417–3432, 2005.

[42] U. Erez and R. Zamir. Achieving $\frac{1}{2}\log(1 + \mathsf{snr})$ on the AWGN channel with lattice encoding and decoding. *IEEE Trans. Inf. Theory*, 50(10):2293–2314, 2004.

[43] A. J. Felstrom and K. S. Zigangirov. Time-varying periodic convolutional codes with low-density parity-check matrix. *IEEE Trans. Inf. Theory*, 45(6):2181–2191, 1999.

[44] A. Fengler, P. Jung, and G. Caire. Sparcs for unsourced random access. arXiv:1901.06234, 2019.

[45] G. D. Forney and G. Ungerboeck. Modulation and coding for linear gaussian channels. *IEEE Trans. Inf. Theory*, 44(6):2384–2415, 1998.

[46] R. G. Gallager. *Information theory and reliable communication*, volume 2. Springer, 1968.

[47] S. Gelfand and M. Pinsker. Coding for a channel with random parameters. *Problems of Control and Information*, 9:19 – 31, January 1980.

[48] B. Girod, A. M. Aaron, S. Rane, and D. Rebollo-Monedero. Distributed video coding. *Proceedings of the IEEE*, 93(1):71 –83, Jan. 2005.

[49] H. Gish and J. Pierce. Asymptotically efficient quantizing. *IEEE Trans. Inf. Theory*, 14(5):676–683, 1968.

[50] A. Goldsmith. *Wireless communications*. Cambridge University Press, 2005.

[51] A. Greig and R. Venkataramanan. Techniques for improving the finite length performance of sparse superposition codes. *IEEE Transactions on Communications*, 66(3):905–917, 2018.

[52] A. Guillén i Fàbregas, A. Martinez, and G. Caire. *Bit-interleaved coded modulation*. Now Publishers Inc, 2008.

[53] A. Gupta and S. Verdù. Nonlinear sparse-graph codes for lossy compression. *IEEE Trans. Inf. Theory*, 55(5):1961 –1975, May 2009.

[54] A. Gupta, S. Verdù, and T. Weissman. Rate-distortion in near-linear time. In *Proc. IEEE Int. Symp. on Inf. Theory*, 2008.

[55] V. Guruswami and P. Xia. Polar codes: Speed of polarization and polynomial gap to capacity. *IEEE Trans. Inf. Theory*, 61(1):3–16, Jan. 2015.

[56] P. Hall. On the rate of convergence of normal extremes. *Journal of Applied Probability*, 16(2):433–439, 1979.

[57] J. Hamkins and K. Zeger. Gaussian source coding with spherical codes. *IEEE Trans. Inf. Theory,*, 48(11):2980–2989, Nov. 2002.

[58] C. Herzet, A. Drémeau, and C. Soussen. Relaxed recovery conditions for omp/ols by exploiting both coherence and decay. *IEEE Trans. Inf. Theory*, 62(1):459–470, 2016.

[59] K. Hsieh, C. Rush, and R. Venkataramanan. Spatially coupled sparse regression codes: Design and state evolution analysis. In *Proc. IEEE Int. Symp. Inf. Theory*. IEEE, 2018.

[60] S. Ihara and M. Kubo. Error exponent for coding of memoryless Gaussian sources with a fidelity criterion. *IEICE Trans. Fundamentals*, E83-A(10), Oct. 2000.

[61] A. Ingber and Y. Kochman. The dispersion of lossy source coding. In *Data Compression Conference (DCC)*, pages 53 –62, March 2011.

[62] S. Jalali and T. Weissman. Rate-distortion via Markov Chain Monte Carlo. In *Proc. IEEE Int. Symp. on Inf. Theory*, 2010.

[63] S. Janson. *Random Graphs*. Wiley, 2000.

[64] A. Joseph. *Achieving information-theoretic limits with high-dimensional regression*. PhD thesis, Yale University, 2012.

[65] A. Joseph and A. R. Barron. Fast sparse superposition codes have near exponential error probability for $R < \mathcal{C}$. *IEEE Trans. Inf. Theory*, 60(2):919–942, Feb. 2014.

[66] I. Kontoyiannis. An implementable lossy version of the Lempel-Ziv algorithm-I: Optimality for memoryless sources. *IEEE Trans. Inf. Theory*, 45(7):2293 –2305, nov 1999.

[67] I. Kontoyiannis and C. Gioran. Efficient random codebooks and databases for lossy compression in near-linear time. In *IEEE Inf. Theory Workshop*, pages 236 –240, 2009.

[68] I. Kontoyiannis, K. Rad, and S. Gitzenis. Sparse superposition codes for Gaussian vector quantization. In *IEEE Inf. Theory Workshop*, page 1, 2010.

[69] S. Korada and R. Urbanke. Polar codes are optimal for lossy source coding. *IEEE Trans. Inf. Theory*, 56(4):1751 –1768, April 2010.

[70] S. Korada and R. Urbanke. Polar codes for slepian-wolf, wyner-ziv, and gelfand-pinsker. In *IEEE Inf. Theory Workshop*, Jan. 2010.

[71] V. Kostina and S. Verdú. Fixed-length lossy compression in the finite blocklength regime. *IEEE Trans. on Inf. Theory*, 58(6):3309–3338, 2012.

[72] F. Krzakala, M. Mézard, F. Sausset, Y. Sun, and L. Zdeborová. Probabilistic reconstruction in compressed sensing: algorithms, phase diagrams, and threshold achieving matrices. *Journal of Statistical Mechanics: Theory and Experiment*, (8), 2012.

[73] S. Kudekar and H. D. Pfister. The effect of spatial coupling on compressive sensing. In *Proc. 48th Annual Allerton Conference on Communication, Control, and Computing*, pages 347–353, 2010.

[74] S. Kudekar, T. Richardson, and R. L. Urbanke. Spatially coupled ensembles universally achieve capacity under belief propagation. *IEEE Trans. Inf. Theory*, 59(12):7761–7813, December 2013.

[75] S. Kudekar, T. J. Richardson, and R. L. Urbanke. Threshold saturation via spatial coupling: Why convolutional LDPC ensembles perform so well over the BEC. *IEEE Trans. Inf. Theory*, 57(2):803–834, 2011.

[76] S. Kumar, A. J. Young, N. Macris, and H. D. Pfister. Threshold saturation for spatially coupled LDPC and LDGM codes on BMS channels. *IEEE Trans. Inf. Theory*, 60(12):7389–7415, 2014.

[77] A. Lapidoth. On the role of mismatch in rate distortion theory. *IEEE Trans. Inf. Theory*, 43(1):38 –47, Jan 1997.

[78] M. Lentmaier, A. Sridharan, D. J. Costello, and K. S. Zigangirov. Iterative decoding threshold analysis for LDPC convolutional codes. *IEEE Trans. Inf. Thy*, 56(10):5274–5289, 2010.

[79] S. Liang, J. Ma, and L. Ping. Clipping can improve the performance of spatially coupled sparse superposition codes. *IEEE Commun. Letters*, 21(12):2578–2581, Dec. 2017.

[80] S. Lin and D. J. Costello. *Error control coding*, volume 2. Prentice Hall Englewood Cliffs, 2004.

[81] K. Marton. Error exponent for source coding with a fidelity criterion. *IEEE Trans. Inf. Theory*, 20(2):197 – 199, Mar 1974.

[82] D. G. Mitchell, M. Lentmaier, and D. J. Costello. Spatially coupled ldpc codes constructed from protographs. *IEEE Trans. Inf. Theory*, 61(9):4866–4889, 2015.

[83] A. Montanari. Graphical models concepts in compressed sensing. In Y. C. Eldar and G. Ku-tyniok, editors, *Compressed Sensing*, pages 394–438. Cambridge University Press, 2012.

[84] P. Moulin and R. Koetter. Data-hiding codes. *Proc. IEEE*, 93(12):2083 – 2126, Dec. 2005.

[85] A. No and T. Weissman. Rateless lossy compression via the extremes. *IEEE Trans. Inf. Theory*, 62(10):5484–5495, 2016.

[86] J. O'Sullivan, P. Moulin, and J. Ettinger. Information theoretic analysis of steganography. In *IEEE Int. Symp. on Inf. Theory*, 1998.

[87] Y. Polyanskiy, H. V. Poor, and S. Verdú. Channel coding rate in the finite blocklength regime. *IEEE Trans. Inf. Theory*, 56(5):2307–2359, 2010.

[88] S. Pradhan, J. Chou, and K. Ramchandran. Duality between source coding and channel coding and its extension to the side information case. *IEEE Trans. Inf. Theory*, 49(5):1181 – 1203, May 2003.

[89] S. Pradhan and K. Ramchandran. Distributed source coding using syndromes (DISCUS): design and construction. *IEEE Trans. Inf. Theory*, 49(3):626 – 643, Mar 2003.

[90] R. Puri, A. Majumdar, and K. Ramchandran. Prism: A video coding paradigm with motion estimation at the decoder. *IEEE Trans. Image Process.*, 16(10):2436 –2448, Oct. 2007.

[91] S. Rangan. Generalized approximate message passing for estimation with random linear mixing. In *Proc. IEEE Int. Symp. Inf. Theory*, pages 2168–2172, 2011.

[92] D. Rebollo-Monedero, R. Zhang, and B. Girod. Design of optimal quantizers for distributed source coding. In *Data Compression Conference*, pages 13 – 22, March 2003.

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

[94] B. Rimoldi. Successive refinement of information: characterization of the achievable rates. *IEEE Trans. Inf. Theory*, 40(1):253 –259, Jan 1994.

[95] C. Rush, A. Greig, and R. Venkataramanan. Capacity-achieving sparse superposition codes via approximate message passing decoding. *IEEE Trans. Inf. Theory*, 63(3):1476–1500, March 2017.

[96] C. Rush, K. Hsieh, and R. Venkataramanan. Capacity-achieving sparse regression codes via spatial coupling. In *Proc. IEEE Inf. Theory Workshop*, 2018.

[97] C. Rush and R. Venkataramanan. The error probability of sparse superposition codes with approximate message passing decoding. Arxiv:1712.06866, 2017.

[98] D. Sakrison. A geometric treatment of the source encoding of a Gaussian random variable. *IEEE Trans. Inf. Theory*, 14(3):481 – 486, May 1968.

[99] D. Sakrison. The rate distortion function for a class of sources. *Information and Control*, 15(2):165 – 195, 1969.

[100] D. Sakrison. The rate of a class of random processes. *IEEE Trans. Inf. Theory*, 16(1):10 – 16, Jan 1970.

[101] J. L. Shanks. Computation of the Fast Walsh-Fourier transform. *IEEE Trans. Comput.*, 18(5):457–459, May 1969.

[102] N. Sommer, M. Feder, and O. Shalvi. Low-density lattice codes. *IEEE Trans. on Inf. Theory*, 54(4):1561–1585, April 2008.

[103] Q. Spencer, C. Peel, A. Swindlehurst, and M. Haardt. An introduction to the multi-user mimo downlink. *IEEE Commun. Mag.*, 42(10):60 – 67, Oct. 2004.

[104] Y. Sun, Y. Yang, A. Liveris, V. Stankovic, and Z. Xiong. Near-capacity dirty-paper code design: A source-channel coding approach. *IEEE Trans. Inf. Theory*, 55(7):3013 –3031, July 2009.

[105] Y. Takeishi, M. Kawakita, and J. Takeuchi. Least squares superposition codes with Bernoulli dictionary are still reliable at rates up to capacity. *IEEE Trans. Inf. Theory*, 60:2737–2750, 2014.

[106] Y. Takeishi and J. Takeuchi. An improved upper bound on block error probability of least squares superposition codes with unbiased bernoulli dictionary. In *IEEE Int. Symp. on Inf. Theory*, pages 1168–1172, 2016.

[107] S. ten Brink. Convergence of iterative decoding. *Electronics letters*, 35(13):1117–1119, 1999.

[108] S. ten Brink, G. Kramer, and A. Ashikhmin. Design of low-density parity-check codes for modulation and detection. *IEEE Trans. Commun.*, 52(4):670–678, 2004.

[109] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[110] R. Tibshirani, M. Wainwright, and T. Hastie. *Statistical learning with sparsity: the LASSO and generalizations*. Chapman and Hall/CRC, 2015.

[111] D. Tse and P. Viswanath. *Fundamentals of wireless communication*. Cambridge University Press, 2005.

[112] R. Venkataramanan, A. Joseph, and S. Tatikonda. Lossy compression via sparse linear regression: Performance under minimum-distance encoding. *IEEE Trans. Inf. Thy*, 60(6):3254–3264, June 2014.

[113] R. Venkataramanan, T. Sarkar, and S. Tatikonda. Lossy compression via sparse linear regression: Computationally efficient encoding and decoding. *IEEE Trans. Inf. Theory*, 60(6):3265–3278, June 2014.

[114] R. Venkataramanan and S. Tatikonda. Sparse regression codes for multi-terminal source and channel coding. In *50th Allerton Conf. on Commun., Control, and Computing*, 2012.

[115] R. Venkataramanan and S. Tatikonda. The rate-distortion function and excess-distortion exponent of sparse regression codes with optimal encoding. *IEEE Trans. Inf. Theory*, 63(8):5228–5243, August 2017.

[116] M. Wainwright, E. Maneva, and E. Martinian. Lossy source compression using low-density generator matrix codes: Analysis and algorithms. *IEEE Trans. Inf. Theory*, 56(3):1351 –1368, 2010.

[117] A. D. Wyner and J. Ziv. The rate-distortion function for source coding with side information at the decoder. *IEEE Trans. Inf. Theory*, 22:1 – 10, January 1976.

[118] Y. Yan, L. Liu, C. Ling, and X. Wu. Construction of capacity-achieving lattice codes: Polar lattices. arXiv:1411.0187, 2014.

[119] Y. Yang, S. Cheng, Z. Xiong, and W. Zhao. Wyner-Ziv coding based on TCQ and LDPC codes. *IEEE Trans. Commun.*, 57(2):376 –387, Feb. 2009.

[120] A. Yedla, Y.-Y. Jian, P. S. Nguyen, and H. D. Pfister. A simple proof of Maxwell saturation for coupled scalar recursions. *IEEE Trans. Inf. Theory*, 60(11):6943–6965, 2014.

[121] R. Zamir. *Lattice Coding for Signals and Networks: A Structured Coding Approach to Quantization, Modulation, and Multiuser Information Theory.* Cambridge University Press, 2014.

[122] R. Zamir, S. Shamai, and U. Erez. Nested linear/lattice codes for structured multiterminal binning. *IEEE Trans. Inf. Theory*, 48(6):1250 –1276, June 2002.