

# The Turbo Decoding Algorithm and Its Phase Trajectories

Dakshi Agrawal, *Member, IEEE*, and Alexander Vardy, *Fellow, IEEE*

**Abstract**—This paper analyzes phase trajectories and fixed points of the turbo decoding algorithm as a function of signal-to-noise ratio (SNR). By exploiting the large length of turbo codes, the turbo decoding algorithm is treated as a single-parameter dynamical system, parameterized (approximately) by the SNR. This parameterization, along with extensive simulations at practical SNRs and asymptotic analysis as SNR goes to zero and infinity, is used to subdivide the entire SNR range into three regions with the “waterfall region” in the middle. The turbo decoding algorithm has distinctive phase trajectories and convergence properties in these three SNR regions. This paper also investigates existence and properties of fixed points in these SNR regions.

The main fixed points of the turbo decoding algorithm are classified into two categories. In a wide range of SNRs (corresponding to bit-error rates less than  $10^{-1}$ ), the decoding algorithm has an “unequivocal” fixed points which correspond to mostly correct decisions on the information bits. Within this range, toward the lower values of SNR, there is another fixed point which corresponds to many erroneous decision on the information bits. Fixed points of this type are referred to as “indecisive” fixed points. It turns out that the indecisive fixed points bifurcate and disappear for SNRs in the waterfall region. This paper associates the qualitative transition of phase trajectories in the waterfall region to the bifurcation of indecisive fixed points. These bifurcations also explain empirically observed quasi-periodic and periodic phase trajectories of the turbo decoding algorithm.

**Index Terms**—Bifurcation theory, dynamical systems, phase trajectories, turbo decoding algorithm.

## I. INTRODUCTION

**I**N their seminal paper [4], Berrou, Glavieux, and Thitimajshima presented a rate-1/2 turbo code which achieved bit-error rate of  $10^{-5}$  over a binary-input additive white Gaussian noise (AWGN) channel at a signal-to-noise ratio (SNR) of 0.7 dB. This is only 0.51 dB away from the Shannon limit for reliable transmission over this channel. Ever since, turbo codes and the associated iterative decoding algorithm have become a focus of intense scrutiny. The structure of turbo codes has been illuminated [2], [11], extended [6], and the decoding algorithm has been simplified [8].

Manuscript received January 3, 2000; revised August 1, 2000. This work was supported by the National Science Foundation and by the David and Lucille Packard Foundation.

D. Agrawal is with T. J. Watson Research Center, IBM Corporation, Hawthorne, NY 10532 USA (e-mail: agrawal@us.ibm.com).

A. Vardy is with the Department of Electrical Engineering, University of California at San Diego, La Jolla, CA 92093 USA (e-mail: vardy@ece.ucsd.edu).

Communicated by B. J. Frey, Guest Editor.

Publisher Item Identifier S 0018-9448(01)00733-7.

Despite these efforts, the performance and nature of the turbo decoding algorithm are still poorly understood. One impediment has been the lack of a mathematical framework for analyzing the turbo decoding algorithm. Often, iterative algorithms are analyzed by identifying a meaningful cost function which is monotonic with respect to the number of iterations. For turbo decoding, no such cost function is known. Similarly, there is no intuitive explanation of the *ad hoc* concept of “extrinsic information” which is passed between the constituent decoders. These factors limit the utility of traditional analysis techniques when applied to turbo decoding.

In a pioneering paper [12], Richardson formalized the turbo decoding algorithm as a discrete dynamical system. Using this approach, he showed the existence of fixed points (also shown by Duan and Rimoldi [7]), and he gave a set of sufficient conditions for the uniqueness and the stability of such fixed points. This approach also yielded a qualitative measure of proximity between turbo decoding and maximum-likelihood decoding.

As a dynamical system, the turbo decoding algorithm can have a variety of phase trajectories. A phase trajectory may converge to a fixed point, reach a well-defined invariant set, or simply wander in the high-dimensional space of extrinsic information. Previously, there has been little insight into the characteristics of these phase trajectories. For example, in many cases, the turbo decoding algorithm does not converge after a fixed number (say 18) of iterations. Is it possible that in the majority of such cases the decoding algorithm actually converges, albeit only after a large number of iterations? Or is the opposite true, that in the majority of such cases, the decoding will never converge. Can these two possibilities be predicted by examining a phase trajectory for a few iterations? These are some of the basic questions addressed here and the focus of this paper is on analyzing the phase trajectories as a function of SNR.

The rest of this paper is organized as follows. In Section II, we establish notation and various conventions. In this regard, we follow Richardson [12] as closely as possible. We establish notation for various density functions that are used to describe the turbo decoding algorithm as a dynamical system and discuss their special properties. We also describe classical turbo codes and the turbo decoding algorithm.

In order to make the paper self-contained, Section III outlines Richardson’s formulation of the turbo decoding algorithm as a dynamical system [12]. This dynamical system iterates on the extrinsic information provided by the two constituent decoders and is parameterized by the channel likelihood ratios of the codeword bits. Since turbo codes have large lengths, it implies that the turbo decoding algorithm is a high-dimensional

dynamical system parameterized by a large number of parameters.

Section IV is devoted to the characterization of the turbo decoding algorithm at asymptotic SNRs. It is shown that at asymptotically low SNRs, with high probability, the turbo decoding algorithm has a unique fixed point that corresponds to many erroneous decisions on the information bits. On the other hand, at asymptotically high SNRs, we show that, with high probability, there are fixed points that correspond to correct decisions on the information bits. Furthermore, at asymptotically high SNRs, the turbo decoding algorithm can converge to only one of these fixed points.

The extensive simulations described in Section V reveal a relationship between fixed points at practical SNRs and fixed points shown to exist at asymptotic SNRs. In particular, simulations show that at low SNRs, before the waterfall region, the turbo decoding algorithm often converges to an “indecisive” fixed point. This fixed point has characteristics that are predicted by the analysis for fixed points at asymptotically low SNRs. On the other hand, at slightly higher SNRs, after the waterfall region, the turbo decoding algorithm converges to an “unequivocal” fixed point. The likelihood ratios of the information bits implied by this fixed point are extreme (either very small or very large), and often the likelihood ratios correspond to correct decisions. These fixed points at relatively moderate SNRs have the same characteristics as the fixed points at asymptotically high SNRs. Finally, simulation results show that contrary to the convergence for SNRs outside the waterfall region, if the SNR is within the waterfall region, the turbo decoding algorithm may or may not converge. In this range, its phase trajectory may become quasi-periodic or periodic.

In Section VI, we treat the turbo decoding algorithm as a dynamical system parameterized by a single parameter that closely approximates the SNR. By varying this parameter, we were able to analyze the turbo decoding algorithm as a function of SNR. In each instance of the turbo decoding algorithm that we analyzed, an unequivocal fixed point was found in a wide range of SNRs (corresponding to bit-error rates less than  $10^{-1}$ ). However, at low SNRs, before the waterfall region, the decoding algorithm was unable to find this fixed point and it converged to an indecisive fixed point. In each instance, as the SNR was increased, this indecisive fixed point became less stable. Ultimately, the indecisive fixed point bifurcated, and it either disappeared or resulted in an attracting invariant set in its neighborhood. On increasing the SNR further, these invariant sets also either disappear or become nonattracting. Once the indecisive fixed point and its associated invariant set disappeared or became nonattracting, in each instance, the turbo decoding algorithm converged to an unequivocal fixed point. In most cases, this disappearance or nonattraction of indecisive fixed points and their invariant sets takes place in the waterfall region, and as a result, the performance of turbo decoding improved dramatically after the waterfall region.

Section VII considers an application of the theory developed here. By exploiting the easily distinguishable characteristics of the indecisive and unequivocal fixed points, we propose a new stopping criterion. This stopping criterion significantly reduces the average number of iterations while keeping the bit-error rate

almost the same. We conclude in Section VIII with a discussion of our results.

## II. PRELIMINARIES

In this section, we establish notation and conventions used in this paper. To that end, we first describe various density functions defined on the set of all binary strings of a fixed length. This is followed by a description of the classical turbo codes and associated iterative decoding algorithm.

### A. Probability Densities

Let  $\mathcal{H}$  be the set of all ordered binary strings of length  $n$ . We use  $\mathbf{b}^0, \mathbf{b}^1, \dots, \mathbf{b}^{2^n-1}$  to denote the elements of  $\mathcal{H}$  as column vectors in a special order. With this ordering,

$$\begin{aligned} \mathbf{b}^0 &= (0, 0, \dots, 0)^T \\ \mathbf{b}^1 &= (1, 0, \dots, 0)^T \\ \mathbf{b}^2 &= (0, 1, 0, \dots, 0)^T, \dots, \mathbf{b}^n = (0, \dots, 0, 1)^T \\ \mathbf{b}^{n+1} &= (1, 1, 0, \dots, 0)^T, \dots, \mathbf{b}^{2^n-1} = (1, 1, \dots, 1)^T. \end{aligned}$$

That is, the elements of  $\mathcal{H}$  are sorted in the increasing order of Hamming weight, and within each weight class, they are sorted in the reverse lexicographical order. A *density* on  $\mathcal{H}$  is a positive real function defined over  $\mathcal{H}$ . A density  $p$  is referred to as a *probability density* if  $\sum_{\mathbf{b} \in \mathcal{H}} p(\mathbf{b}) = 1$ . Given a density  $f$ , there is a unique positive constant  $\lambda(f) = \sum_{\mathbf{b} \in \mathcal{H}} f(\mathbf{b})$  such that  $\hat{f} = f/\lambda(f)$  is a probability density. A density  $f$  on  $\mathcal{H}$  induces a *probability measure*  $\Pr_f$  on  $\mathcal{P}(\mathcal{H})$ , the set of all subsets of  $\mathcal{H}$ , as follows:

$$\Pr_f(A) := \sum_{\mathbf{b} \in A} \hat{f}(\mathbf{b}) \quad \forall A \in \mathcal{P}(\mathcal{H}).$$

We say that densities  $p$  and  $q$  are equivalent if they determine the same probability density. This is an equivalence relation and divides the set of all densities into equivalence classes. We say that a density  $f$  is normalized with respect to the all-zero binary string  $\mathbf{b}^0 = \mathbf{0}$  if  $f(\mathbf{0}) = 1$ . In each equivalence class, there is a unique probability density and a unique density normalized with respect to  $\mathbf{0}$ . The proportional relationship between these two representatives is given by

$$f(\mathbf{b}) = \frac{\hat{f}(\mathbf{b})}{\hat{f}(\mathbf{b}^0)}, \quad \mathbf{b} \in \mathcal{H}$$

where  $f$  is the density normalized with respect to  $\mathbf{0}$  and  $\hat{f}$  is the equivalent probability density. We will almost always deal with one of these two representatives from each equivalence class. For brevity, a density normalized with respect to  $\mathbf{0}$  will simply be called a *normalized density*.

It is useful to represent densities in the logarithmic domain. Given a density  $f$ , let  $F = \log \circ f$  be its logarithmic representation. We use a lower case letter to denote a density and the corresponding capital letter to denote its representation in the logarithmic domain. Logarithmic representation  $F$  of a density  $f$  is a real-valued function on  $\mathcal{H}$ , taking both positive and negative values. To avoid cumbersome language, we say that  $F$  is a *log-density* on  $\mathcal{H}$ .

A density  $f$ , normalized with respect to  $\mathbf{0}$ , corresponds to a log-density  $F$  with  $F(\mathbf{0}) = 0$ . Let  $\Phi$  denote the set of all log-densities that correspond to the normalized densities, that is,  $F \in \Phi$  if and only if  $F(\mathbf{0}) = 0$ . Since for a log-density  $F \in \Phi$ ,  $F(\mathbf{0})$  is always zero,  $\Phi$  can be directly identified with  $\mathbb{R}^{2^n-1}$ .

Let  $\mathcal{H}_i \subset \mathcal{H}$  be the set of all binary strings whose  $i$ th bit is 1. A density  $f$  is referred to as a *product density* if, according to the induced probability measure  $\Pr_f$ , all bits are independent of each other. More precisely, the density  $f$  is a product density if according to  $\Pr_f$ , the events  $\mathcal{H}_i, i = 1, \dots, n$ , are statistically independent of each other. For a product density  $f$  which is also normalized, we have

$$\begin{aligned} f(\mathbf{b} = b_1 b_2 \dots b_n) &= \frac{\hat{f}(\mathbf{b})}{\hat{f}(\mathbf{0})} \\ &= \prod_{\{i: b_i=1\}} \frac{\Pr_f(\mathcal{H}_i)}{\Pr_f(\mathcal{H}_i^c)} \end{aligned} \quad (1)$$

where the first equality follows since  $f$  is normalized, while the last equality is a result of  $f$  being a product density. It is clear from (1) that for a normalized product density  $f$ ,  $f(\mathbf{b}^i)$ ,  $i = 1, \dots, n$ , is the likelihood ratio of the  $i$ th bit according to the density  $f$

$$f(\mathbf{b}^i) = \frac{\Pr_f(\mathcal{H}_i)}{\Pr_f(\mathcal{H}_i^c)} \quad (2)$$

$$= \frac{\sum_{\{\mathbf{b} \in \mathcal{H}_i\}} f(\mathbf{b})}{\sum_{\{\mathbf{b} \in \mathcal{H}_i^c\}} f(\mathbf{b})}. \quad (3)$$

Using the likelihood ratios of the individual bits, we can recast (1) as

$$f(\mathbf{b} = b_1 b_2 \dots b_n) = \prod_{\{i: b_i=1\}} f(\mathbf{b}^i). \quad (4)$$

We refer to a log-density that corresponds to a product density as a *product log-density*. Let  $\Pi$  be the set of all product log-densities in  $\Phi$ . Using (4), for a product log-density  $F \in \Pi$ , we have

$$F(\mathbf{b} = b_1 b_2 \dots b_n) = \sum_{\{i: b_i=1\}} F(\mathbf{b}^i). \quad (5)$$

Here,  $F(\mathbf{b}^i)$  is the log-likelihood ratio of the  $i$ th bit according to the density  $f$ . Therefore, densities in  $\Pi$  are completely specified by their values on  $\mathbf{b}^1, \mathbf{b}^2, \dots, \mathbf{b}^n$ . Furthermore,  $\Pi$  is an  $n$ -dimensional linear subspace of  $\Phi$ . A basis for  $\Pi$  is given by the columns of a  $2^n \times n$  matrix  $B$ , having  $(\mathbf{b}^i)^T$  as its  $i$ th row

$$B_{i,j} = \mathbf{b}_j^i, \quad i = 0, 1, \dots, 2^n - 1; j = 1, 2, \dots, n. \quad (6)$$

In turbo decoding, normalized product-densities play a central role. We will make frequent use of the fact that a normalized product-density on  $\mathcal{H}$  can be specified by the likelihood ratios of the bits.

We say that two densities  $p$  and  $q$  have the same bitwise marginal distributions if

$$\Pr_p(\mathcal{H}_i) = \Pr_q(\mathcal{H}_i), \quad \text{for } i = 1, \dots, n. \quad (7)$$

For a log-density  $P$ , we define a projection map  $\pi_P: \Pi \rightarrow \Pi$  by setting  $\pi_P(Q)$  to be the unique normalized product log-density that has the same bitwise marginals as  $P+Q$ . Using (2) and (7), we have

$$\begin{aligned} \pi_P(Q)(\mathbf{b}^i) &= \log \frac{\Pr_{\pi_P(Q)}(\mathcal{H}_i)}{\Pr_{\pi_P(Q)}(\mathcal{H}_i^c)} \\ &= \log \frac{\Pr_{P+Q}(\mathcal{H}_i)}{\Pr_{P+Q}(\mathcal{H}_i^c)} \\ &= \log \frac{\sum_{\{\mathbf{b} \in \mathcal{H}_i\}} p(\mathbf{b})q(\mathbf{b})}{\sum_{\{\mathbf{b} \in \mathcal{H}_i^c\}} p(\mathbf{b})q(\mathbf{b})}, \quad \text{for } i = 1, 2, \dots, n. \end{aligned} \quad (8)$$

Since  $\pi_P(Q)$  is a normalized product log-density, it is completely specified by (8). This shows that  $\pi_P(Q)$  is indeed unique, and hence well defined. Note that  $\pi_P(Q)(\mathbf{b}^i)$ ,  $i = 1, 2, \dots, n$ , is the log-likelihood ratio of the  $i$ th bit according to the log-density  $P+Q$ . Later on, this fact will be used when we describe the turbo decoding algorithm as a discrete dynamical system.

### B. Turbo Codes and Iterative Decoding Algorithm

A classical turbo code is a parallel concatenation of two recursive systematic binary convolutional codes [4]. Turbo codes are best described by their encoding operation. The encoder for a turbo code consists of two subencoders, one for each constituent convolutional code. The inputs of subencoders are related through a pseudorandom interleaver. The length of the pseudorandom interleaver, denoted by  $n$ , equals the number of information bits encoded into one turbo codeword. Let  $\mathcal{C}_1$  and  $\mathcal{C}_2$  be the constituent convolutional codes.

Let  $\mathbf{i}$  be the information bit sequence of length  $n$  at the input to the turbo encoder. Before starting the encoding operation, both convolutional encoders are reset to the zero state. The sequence  $\mathbf{i}$  is fed directly to the first convolutional encoder. The second convolutional encoder uses the same information bits in a different sequential order obtained by reordering according to a pseudorandom interleaver. After the information bits, each encoder is fed  $m$  extra bits where  $m$  is the memory length of encoders. These extra bits are selected for each encoder in such a manner that these bits drive their respective encoders back to the zero state, thus *terminating* their trellises. Since the constituent encoders are recursive, sending  $m$  zero bits does not necessarily bring the encoders back to the zero state. Instead, for each encoder, the  $m$  extra bits depend linearly on the information bits, and they can be treated as parity bits generated by that encoder [6].

To increase the rate, parity bits generated by both convolutional encoders are punctured in a suitable pattern [3]. For input information bit sequence  $\mathbf{i}$ , let  $\mathbf{c}_1(\mathbf{i})$ , respectively,  $\mathbf{c}_2(\mathbf{i})$ , be the parity bit sequences produced by the first, respectively, the second, encoder that are left after puncturing. The information bit sequence  $\mathbf{i}$  along with the parity bit sequences  $\mathbf{c}_1(\mathbf{i})$  and  $\mathbf{c}_2(\mathbf{i})$  forms a turbo codeword  $(\mathbf{i}, \mathbf{c}_1(\mathbf{i}), \mathbf{c}_2(\mathbf{i}))$ .

We assume that the turbo code is transmitted over a noisy binary-input memoryless channel. Let  $p(\cdot|b)$ ,  $b = 0$  and  $1$ , be the probability density function of the channel output given that

the input bit was  $b$ . At the receiver, for each input bit  $v$ , a *channel likelihood ratio*

$$\frac{p(\tilde{v}|v=1)}{p(\tilde{v}|v=0)}$$

is computed, where  $\tilde{v}$  is the channel output for the input bit  $v$ . The turbo decoding algorithm computes several likelihood ratios for each information bit. These likelihood ratios correspond to different sets of observations and it is important not to confuse them with each other.

Ideally, we would like to compute the *posterior* probability density  $p(\mathbf{b}|\tilde{\mathbf{z}}, \tilde{\mathbf{c}}_1, \tilde{\mathbf{c}}_2)$ , where  $\mathbf{b} \in \mathcal{H}$ , and  $\tilde{\mathbf{z}}$ ,  $\tilde{\mathbf{c}}_1$ , and  $\tilde{\mathbf{c}}_2$  are the channel outputs corresponding to the input sequences  $\mathbf{i}$ ,  $\mathbf{c}_1(\mathbf{i})$ , and  $\mathbf{c}_2(\mathbf{i})$ , respectively. Let us assume that the input bits are independent of each other and are equally likely to be either 0 or 1. Under this standard assumption, on a memoryless channel,  $p(\mathbf{b}|\tilde{\mathbf{z}}, \tilde{\mathbf{c}}_1, \tilde{\mathbf{c}}_2)$  is equivalent (proportional) to the density

$$p_{ML}(\mathbf{b}) = p(\mathbf{i}|\mathbf{b})p(\tilde{\mathbf{c}}_1|\mathbf{b})p(\tilde{\mathbf{c}}_2|\mathbf{b}).$$

A direct computation of  $p_{ML}$  requires taking both sets of parity bits,  $\mathbf{c}_1(\mathbf{i})$  and  $\mathbf{c}_2(\mathbf{i})$ , simultaneously into account by constructing a joint trellis of two convolutional encoders. This is computationally prohibitive. Instead, the turbo decoding algorithm deals with only one set of parity bits at a time.

A turbo decoder consists of two components: a decoder  $\mathcal{D}_1$  for the convolutional code  $\mathcal{C}_1$  and a decoder  $\mathcal{D}_2$  for the convolutional code  $\mathcal{C}_2$ . These decoders use the Bahl–Cocke–Jelinek–Raviv (BCJR) algorithm [1] to compute *a posteriori* probabilities (APPs) of the information bits, and they can take into account an *a priori* product density on the information bits. Let  $q_1$  be the *a priori* product density used by decoder  $\mathcal{D}_2$  and let  $q_2$  be the *a priori* product density used by decoder  $\mathcal{D}_1$ . These densities are also referred to as the prior densities.

To begin with, prior densities for both decoders are initialized to the uniform density. This is called the *unbiased* initialization of the turbo decoder. The decoding begins with the decoder  $\mathcal{D}_1$  computing the posterior likelihood ratios of the information bits based on the prior density  $q_2$  and the observations  $\tilde{\mathbf{z}}$  and  $\tilde{\mathbf{c}}_1$ . The posterior likelihood ratio of the  $i$ th information bit is given by

$$\frac{\sum_{\mathbf{b} \in \mathcal{H}_i} p(\mathbf{b}|\tilde{\mathbf{z}}, \tilde{\mathbf{c}}_1)}{\sum_{\mathbf{b} \in \mathcal{H}_i^c} p(\mathbf{b}|\tilde{\mathbf{z}}, \tilde{\mathbf{c}}_1)} = \frac{\sum_{\mathbf{b} \in \mathcal{H}_i} p(\mathbf{i}|\mathbf{b})p(\tilde{\mathbf{c}}_1|\mathbf{b})q_2(\mathbf{b})}{\sum_{\mathbf{b} \in \mathcal{H}_i^c} p(\mathbf{i}|\mathbf{b})p(\tilde{\mathbf{c}}_1|\mathbf{b})q_2(\mathbf{b})}. \quad (9)$$

Clearly, this ratio does not change if we take other representatives from the equivalence classes of  $q_2(\mathbf{b})$ ,  $p(\mathbf{i}|\mathbf{b})$ , and  $p(\tilde{\mathbf{c}}_1|\mathbf{b})$ .

Next, the *extrinsic information* for the  $i$ th information bit is obtained by dividing its posterior likelihood ratio computed above by the product of its channel and prior likelihood ratios. This extrinsic information is then passed from the decoder  $\mathcal{D}_1$  to the decoder  $\mathcal{D}_2$  by setting the prior density  $q_1$  in such a way that the likelihood ratios of the information bits according to  $q_1$  equal their extrinsic information, that is, we set

$$\frac{q_1(\mathbf{b}^i)}{q_1(\mathbf{b}^0)} = \frac{\sum_{\mathbf{b} \in \mathcal{H}_i} p(\mathbf{b}|\tilde{\mathbf{z}}, \tilde{\mathbf{c}}_1)}{\sum_{\mathbf{b} \in \mathcal{H}_i^c} p(\mathbf{b}|\tilde{\mathbf{z}}, \tilde{\mathbf{c}}_1)} \frac{p(\mathbf{b}^0|\tilde{\mathbf{z}}) q_2(\mathbf{b}^0)}{p(\mathbf{b}^i|\tilde{\mathbf{z}}) q_2(\mathbf{b}^i)} \quad (10)$$

Recall that since  $q_1$  is a product density, the likelihood ratios  $q_1(\mathbf{b}^i)/q_1(\mathbf{b}^0)$ , for  $i = 1, 2, \dots, n$  determine the density  $q_1$  uniquely.

The second decoder  $\mathcal{D}_2$  uses the prior density  $q_1$  to compute the posterior likelihood ratios of the information bits based on the observations  $\tilde{\mathbf{z}}$  and  $\tilde{\mathbf{c}}_2$

$$\frac{\sum_{\mathbf{b} \in \mathcal{H}_i} p(\mathbf{b}|\tilde{\mathbf{z}}, \tilde{\mathbf{c}}_2)}{\sum_{\mathbf{b} \in \mathcal{H}_i^c} p(\mathbf{b}|\tilde{\mathbf{z}}, \tilde{\mathbf{c}}_2)} = \frac{\sum_{\mathbf{b} \in \mathcal{H}_i} p(\mathbf{i}|\mathbf{b})p(\tilde{\mathbf{c}}_2|\mathbf{b})q_1(\mathbf{b})}{\sum_{\mathbf{b} \in \mathcal{H}_i^c} p(\mathbf{i}|\mathbf{b})p(\tilde{\mathbf{c}}_2|\mathbf{b})q_1(\mathbf{b})}, \quad \text{for } i = 1, 2, \dots, n. \quad (11)$$

Analogous to the decoder  $\mathcal{D}_1$ , the decoder  $\mathcal{D}_2$  also computes extrinsic information for the information bits and passes it to the decoder  $\mathcal{D}_1$  by setting its prior density  $q_2$

$$\frac{q_2(\mathbf{b}^i)}{q_2(\mathbf{b}^0)} = \frac{\sum_{\mathbf{b} \in \mathcal{H}_i} p(\mathbf{b}|\tilde{\mathbf{z}}, \tilde{\mathbf{c}}_2)}{\sum_{\mathbf{b} \in \mathcal{H}_i^c} p(\mathbf{b}|\tilde{\mathbf{z}}, \tilde{\mathbf{c}}_2)} \frac{p(\mathbf{b}^0|\tilde{\mathbf{z}}) q_1(\mathbf{b}^0)}{p(\mathbf{b}^i|\tilde{\mathbf{z}}) q_1(\mathbf{b}^i)}. \quad (12)$$

Next, the decoder  $\mathcal{D}_1$  computes the new posterior likelihood ratios based on its updated prior density  $q_2$ , and accordingly updates the prior density  $q_1$  for the second decoder. The decoding algorithm proceeds iteratively in this manner until a stopping criterion (usually, a predetermined number of iterations) is satisfied. The final decision on information bits takes into account the extrinsic information from both decoders as well as the channel likelihood ratios of information bits. For the  $i$ th information bit, the following ratio is computed:

$$\frac{p(\mathbf{b}^i|\tilde{\mathbf{z}}) q_1(\mathbf{b}^i) q_2(\mathbf{b}^i)}{p(\mathbf{b}^0|\tilde{\mathbf{z}}) q_1(\mathbf{b}^0) q_2(\mathbf{b}^0)}. \quad (13)$$

The  $i$ th information bit is decoded as 1 if this ratio is greater than 1 and decoded as 0 otherwise.

### III. THE TURBO DECODING ALGORITHM AS A DYNAMICAL SYSTEM

The turbo decoding algorithm can be elegantly summarized as a dynamical system using log-densities. Let  $p_0$ ,  $p_1$ , and  $p_2$  be the normalized densities equivalent to  $p(\mathbf{i}|\mathbf{b})$ ,  $p(\tilde{\mathbf{c}}_1|\mathbf{b})$ , and  $p(\tilde{\mathbf{c}}_2|\mathbf{b})$ , respectively. Since (9)–(13) are invariant under the equivalence classes of densities, we can describe the turbo decoding algorithm in terms of normalized densities. Let  $q_1$  and  $q_2$  be the *normalized* prior densities used by the second and the first decoder, respectively. Recall that  $q_1$  and  $q_2$  are initialized to induce the uniform probability distribution on  $\mathcal{H}$ . In the logarithmic domain, this is equivalent to setting  $Q_1 = Q_2 = \mathbf{0}$ .

The first decoder  $\mathcal{D}_1$  uses the normalized density  $p_0 p_1 q_2$  [cf. (9)] to compute posterior likelihood ratios. In the logarithmic domain,  $p_0 p_1 q_2$  corresponds to  $P_0 + P_1 + Q_2$ . Since  $(P_0 + Q_2) \in \Pi$ , it follows from (8) [also see the remark following (8)] that the posterior log-likelihood ratio of the  $i$ th information bit is given by  $\pi_{P_1}(P_0 + Q_2)(\mathbf{b}^i)$ . Next, the extrinsic information contributed by  $\mathcal{D}_1$  for the  $i$ th information bit is obtained by subtracting  $(P_0 + Q_2)(\mathbf{b}^i)$ , the channel, and the prior log-likelihood ratio of the  $i$ th information bit from  $\pi_{P_1}(P_0 + Q_2)(\mathbf{b}^i)$  [cf. (10)]. This extrinsic information is used to specify the prior log-density  $Q_1$ . That is, we set

$$Q_1(\mathbf{b}^i) \leftarrow \pi_{P_1}(P_0 + Q_2)(\mathbf{b}^i) - (P_0 + Q_2)(\mathbf{b}^i), \quad \text{for } i = 1, \dots, n. \quad (14)$$

It follows from (5) that  $Q_1$  is uniquely specified by (14)

$$Q_1 \leftarrow \pi_{P_1}(P_0 + Q_2) - (P_0 + Q_2). \quad (15)$$

The second decoder  $\mathcal{D}_2$  performs a similar operation and computes the modified prior log-density  $Q_2$

$$Q_2 \leftarrow \pi_{P_2}(P_0 + Q_1) - (P_0 + Q_1). \quad (16)$$

The decoding algorithm iteratively performs the operations indicated by (15) and (16) until a stopping criterion is satisfied.

As a discrete dynamical system given by (15) and (16), the turbo decoding algorithm can be parameterized by the log-densities  $P_0, P_1$ , and  $P_2 \in \Phi$ . Starting from an initial point  $(Q_1, Q_2) = (Q_1^{(0)}, Q_2^{(0)})$ , it iterates on variables  $(Q_1, Q_2)$  to produce a *phase trajectory*  $(Q_1^{(1)}, Q_2^{(1)}), (Q_1^{(2)}, Q_2^{(2)}), \dots$ . The log-densities  $P_0, P_1$ , and  $P_2$  are completely specified by the channel likelihood ratios of the codeword bits. Consequently, the turbo decoding algorithm is parameterized by  $n/r$  parameters where  $r$  is the code rate. The iterated variables,  $Q_1$  and  $Q_2$ , are product log-densities, and each of them can be specified by  $n$  log-likelihood ratios. Hence, in the above representation, the turbo decoding algorithm is a dynamical system in  $2n$  variables. As shown in [12], it depends smoothly (infinitely differentiable) on its parameters and variables.

This formulation, first proposed by Richardson [12], uses extrinsic information as variables. This choice of variables is natural since extrinsic information is updated and passed between constituent decoders at each iteration. However, note that  $Q_2^{(t)}$  can be specified solely in terms of  $Q_1^{(t)}$ , given the parameters  $P_0, P_1$ , and  $P_2$ . Therefore, the above representation is redundant. In Appendix B, we present a slightly different formulation which eliminates this redundancy.

#### IV. FIXED POINTS AT ASYMPTOTIC SNRS

The turbo decoding algorithm is a complex dynamical system not readily amenable to analysis. In order to simplify the analysis and gain some insight into its operation, we examine it at asymptotically low and asymptotically high SNRs.

In the rest of this paper, we focus on classical turbo codes transmitted over an AWGN channel using binary phase-shift keying (BPSK) modulation. Let  $\mathbf{s}(\mathbf{b})$  be the Euclidean-space representation of the binary string  $\mathbf{b}$  under the BPSK map, and let  $\mathbf{s}_1 = \mathbf{s} \circ \mathbf{c}_1$  and  $\mathbf{s}_2 = \mathbf{s} \circ \mathbf{c}_2$ , where  $\circ$  denotes composition of two functions. Without loss of generality, in the rest of this section we consider the all-zero information sequence  $\mathbf{b}^0$ . Denote the corresponding transmitted vector by  $(\mathbf{x}, \mathbf{y}, \mathbf{z}) : (\mathbf{s}(\mathbf{b}^0), \mathbf{s}_1(\mathbf{b}^0), \mathbf{s}_2(\mathbf{b}^0))$  and denote the corresponding received vector by  $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\mathbf{z}})$ . The normalized posterior density  $p_1$ , induced on the information bits by  $\tilde{\mathbf{y}}$ , is given by

$$p_1(\mathbf{b}) = \frac{\Pr(\mathbf{b}|\tilde{\mathbf{y}})}{\Pr(\mathbf{b}^0|\tilde{\mathbf{y}})} = \exp\left(\frac{\|\tilde{\mathbf{y}} - \mathbf{s}_1(\mathbf{b}^0)\|^2 - \|\tilde{\mathbf{y}} - \mathbf{s}_1(\mathbf{b})\|^2}{2\sigma^2}\right), \quad \mathbf{b} \in \mathcal{H}. \quad (17)$$

Here,  $\sigma^2$  is the noise variance and  $\|\cdot\|^2$  denotes the squared Euclidean distance in  $\mathbb{R}^{l_1}$ ,  $l_1$  being the number of parity bits

contributed by the first convolutional code for each codeword. The corresponding normalized log-density  $P_1$  is given by

$$P_1(\mathbf{b}) = \left(\frac{\|\tilde{\mathbf{y}} - \mathbf{s}_1(\mathbf{b}^0)\|^2 - \|\tilde{\mathbf{y}} - \mathbf{s}_1(\mathbf{b})\|^2}{2\sigma^2}\right) = -\frac{2}{\sigma^2} \sum_{\{j: s_{1j}(\mathbf{b}^0) \neq s_{1j}(\mathbf{b})\}} \tilde{y}_j. \quad (18)$$

Here,  $s_{1j}$  and  $\tilde{y}_j$  are the  $j$ th components of  $\mathbf{s}_1$  and  $\tilde{\mathbf{y}}$ , respectively. The vector  $\tilde{\mathbf{y}}$  is an  $l_1$ -dimensional Gaussian random vector with mean  $\mathbf{s}_1(\mathbf{b}^0)$  and covariance matrix  $\sigma^2 \mathbf{I}_{l_1 \times l_1}$ . It follows that  $P_1(\mathbf{b})$  is a Gaussian random variable with mean  $-\frac{2}{\sigma^2} \text{wt}(\mathbf{c}_1(\mathbf{b}))$  and variance  $\frac{4}{\sigma^2} \text{wt}(\mathbf{c}_1(\mathbf{b}))$

$$P_1(\mathbf{b}) \sim \mathcal{N}\left(-\frac{2}{\sigma^2} \text{wt}(\mathbf{c}_1(\mathbf{b})), \frac{4}{\sigma^2} \text{wt}(\mathbf{c}_1(\mathbf{b}))\right). \quad (19)$$

Here,  $\text{wt}(\cdot)$  denotes the Hamming weight of its argument. In a similar manner, log-densities  $P_0$  and  $P_2$  are induced by the received vectors  $\tilde{\mathbf{x}}$  and  $\tilde{\mathbf{z}}$ , respectively. Specifically,  $P_0(\mathbf{b})$  is a Gaussian random variable with mean  $-\frac{2}{\sigma^2} \text{wt}(\mathbf{b})$  and variance  $\frac{4}{\sigma^2} \text{wt}(\mathbf{b})$ .

#### A. Unique Fixed Point at Asymptotically Low SNRs

In this subsection, we prove that it is highly probable that at asymptotically low SNRs, the turbo decoding algorithm has a unique fixed point. In order to prove this assertion, we use a set of sufficient conditions on  $P_0, P_1$ , and  $P_2$ , provided by Richardson [12], for the uniqueness of the fixed point. The following theorem paraphrases these sufficient conditions in terms of an open set  $\mathcal{U}$ .

*Theorem 1:* There exists an open set  $\mathcal{U} \subset \Pi \times \Phi \times \Phi$  containing  $\Pi \times \Pi \times \Pi$  such that for  $(P_0, P_1, P_2) \in \mathcal{U}$

- 1) the turbo decoding algorithm has a *unique* fixed point  $(Q_1^*, Q_2^*)$ ;
- 2) the fixed point  $(Q_1^*, Q_2^*)$  is *stable*;
- 3) the fixed point  $(Q_1^*, Q_2^*)$  is a *continuous* function of  $(P_0, P_1, P_2) \in \mathcal{U}$ .

Recall that a normalized log-density on  $\mathcal{H}$  can be considered as a point in  $\mathbb{R}^{2^n - 1}$ , and the set of all normalized log-densities  $\Phi$  can be identified with  $\mathbb{R}^{2^n - 1}$ . In Theorem 1, the set  $\mathcal{U}$  is open in the topology induced on  $\Pi \times \Phi \times \Phi$  by its identification as a subset of  $\mathbb{R}^{2^n - 1} \times \mathbb{R}^{2^n - 1} \times \mathbb{R}^{2^n - 1}$ .

Intuitively, at asymptotically low SNRs, the channel output  $\tilde{\mathbf{y}}$  provides very little information about the transmitted information sequence  $\mathbf{x}$ . More precisely, as  $\sigma \rightarrow \infty$ ,  $P_1$  converges to the uniform log-density  $\mathbf{0}$ . Similarly,  $P_0$  and  $P_2$  also converge to  $\mathbf{0}$ . Since the log-density  $\mathbf{0}$  is a product log-density and the set  $\Pi \times \Pi \times \Pi$  is contained in the set  $\mathcal{U}$ , it follows that  $(\mathbf{0}, \mathbf{0}, \mathbf{0})$  belongs to the set  $\mathcal{U}$ . We use Theorem 1 and the following lemma, which proves the convergence of  $P_0, P_1$ , and  $P_2$  to  $\mathbf{0}$  to show that at asymptotically low SNRs, the turbo decoding algorithm has a unique fixed point with high probability.

*Lemma 1:* In the setup assumed above, given an  $\epsilon$  between 0 and 1, there exists a  $\hat{\sigma}(\epsilon) > 0$  such that for  $\sigma > \hat{\sigma}(\epsilon)$

$$(\|P_i - \mathbf{0}\| \geq \epsilon) < \epsilon, \quad \text{for } i = 0, 1, \text{ and } 2.$$

*Proof:* First consider the log-density  $P_1$

$$\begin{aligned} E[\|P_1\|^2] &= \sum_{\mathbf{b} \in \mathcal{H}} E[P_1^2(\mathbf{b})] = \sum_{\mathbf{b} \in \mathcal{H}} \text{Var}[P_1(\mathbf{b})] + E^2[P_1(\mathbf{b})] \\ &= \sum_{\mathbf{b} \in \mathcal{H}} \frac{4}{\sigma^2} \text{wt}(\mathbf{c}_1(\mathbf{b})) + \frac{4}{\sigma^4} (\text{wt}(\mathbf{c}_1(\mathbf{b})))^2. \end{aligned} \quad (20)$$

Since the sum in (20) is over a finite number of terms, as  $\sigma^2 \rightarrow \infty$ ,  $E[\|P_1\|^2] \rightarrow 0$ . Therefore, we can choose a  $\sigma_1(\epsilon)$ , such that for  $\sigma > \sigma_1(\epsilon)$ ,  $E[\|P_1\|^2] < \epsilon^3$ . Next we use Chebychev's inequality

$$\Pr(\|P_1 - \mathbf{0}\| \geq \epsilon) \leq \frac{E[\|P_1 - \mathbf{0}\|^2]}{\epsilon^2} < \epsilon, \quad \text{for } \sigma > \sigma_1(\epsilon).$$

A similar argument for  $P_0$  and  $P_2$  shows that there exist  $\sigma_0(\epsilon)$  and  $\sigma_2(\epsilon)$ , such that

$$\begin{aligned} \Pr(\|P_0 - \mathbf{0}\| \geq \epsilon) &< \epsilon, & \text{for } \sigma > \sigma_0(\epsilon) \\ \Pr(\|P_2 - \mathbf{0}\| \geq \epsilon) &< \epsilon, & \text{for } \sigma > \sigma_2(\epsilon). \end{aligned}$$

Now set  $\hat{\sigma}(\epsilon) = \max(\sigma_0(\epsilon), \sigma_1(\epsilon), \sigma_2(\epsilon))$ .  $\square$

*Theorem 2:* Given any  $\epsilon$  between 0 and 1, we can choose a  $\sigma(\epsilon) > 0$  such that for  $\sigma > \sigma(\epsilon)$ , the turbo decoding algorithm has a unique fixed point with probability more than  $1 - \epsilon$ .

*Proof:* The log-density  $\mathbf{0}$  belongs to the set  $\Pi$ . Therefore, by Theorem 1, the triplet  $(\mathbf{0}, \mathbf{0}, \mathbf{0})$  belongs to the set  $\mathcal{U}$ . Since the set  $\mathcal{U}$  is open, there exists a  $\delta > 0$ , such that if  $\|P_0 - \mathbf{0}\| < \delta$ ,  $\|P_1 - \mathbf{0}\| < \delta$ , and  $\|P_2 - \mathbf{0}\| < \delta$ , then the triplet  $(P_0, P_1, P_2) \in \mathcal{U}$ . Now it follows that

$$\begin{aligned} \Pr((P_0, P_1, P_2) \in \mathcal{U}) &\geq \Pr\left(\bigcap_{i=0}^2 \|P_i - \mathbf{0}\| < \delta\right) \\ &= \prod_{i=0}^2 \Pr(\|P_i - \mathbf{0}\| < \delta) \\ &= \prod_{i=0}^2 [1 - \Pr(\|P_i - \mathbf{0}\| \geq \delta)] \\ &> (1 - \delta)^3, \quad \text{for } \sigma > \hat{\sigma}(\delta). \end{aligned} \quad (21)$$

Here, we used Lemma 1 for the last inequality. Now choose  $\delta$  small enough so that  $(1 - \delta)^3 \geq 1 - \epsilon$ .  $\square$

Theorem 2 shows that by choosing a low enough SNR, the probability of the turbo decoding algorithm having a unique fixed point can be made arbitrarily close to 1. We can draw further conclusions about the properties of this unique fixed point. For the parameter triplet  $(\mathbf{0}, \mathbf{0}, \mathbf{0})$ , the turbo decoding algorithm obviously converges to the fixed point  $(\mathbf{0}, \mathbf{0})$  in a single iteration. Since, by Theorem 1, the unique fixed point is a continuous function of the parameters, we expect that at low SNRs, the unique fixed point  $(Q_1^*, Q_2^*) \approx (\mathbf{0}, \mathbf{0})$  and the unbiased initialization  $(\mathbf{0}, \mathbf{0})$  will be within the domain of attraction of this fixed point. In this case,  $P_0(\mathbf{b}^i) + Q_1^*(\mathbf{b}^i) + Q_2^*(\mathbf{b}^i) \approx P_0(\mathbf{b}^i)$  and the bit-error rate of the turbo decoding will be close to the

error rate of the channel. In Section V, this will be confirmed by simulations.

### B. Fixed Points at Asymptotically High SNRs

For asymptotically high SNRs, we conjecture that the situation is analogous to the one shown here for asymptotically low SNRs. Unfortunately, at this time, we have only a partial proof of this conjecture. We will show that it is highly probable that at asymptotically high SNRs, the turbo decoding algorithm has fixed points that correspond to the correct decisions on the information bits. However, the uniqueness of such fixed points cannot be established at this time.

We start by defining four new parameters that depend on the log-densities  $P_0, P_1$ , and  $P_2$

$$\alpha_1 = \max_{i=1,2,\dots,n} \sum_{\substack{\mathbf{b} \in \mathcal{H}_i \\ \mathbf{b} \neq \mathbf{b}^i}} p_1(\mathbf{b}) \prod_{\substack{j: \mathbf{b} \in \mathcal{H}_j \\ j \neq i}} p_0(\mathbf{b}^j) \quad (22)$$

$$\alpha_2 = \max_{i=1,2,\dots,n} \sum_{\substack{\mathbf{b} \in \mathcal{H}_i \\ \mathbf{b} \neq \mathbf{b}^i}} p_2(\mathbf{b}) \prod_{\substack{j: \mathbf{b} \in \mathcal{H}_j \\ j \neq i}} p_0(\mathbf{b}^j) \quad (23)$$

$$\hat{p}_1 = \max_{i=1,2,\dots,n} p_1(\mathbf{b}^i) \quad (24)$$

$$\hat{p}_2 = \max_{i=1,2,\dots,n} p_2(\mathbf{b}^i). \quad (25)$$

Under the assumption that the all-zero codeword was transmitted and by using the distributions of  $P_0, P_1$ , and  $P_2$ , described in the beginning of this section, it can be shown that as the SNR increases, the above four parameters converge to zero in probability. In other words, by choosing the SNR large enough, the probability of these parameters being greater than  $\epsilon > 0$  can be made smaller than  $\epsilon$ .

Let  $\hat{q}_1^{(l)}$  be an upper bound on the extrinsic information from the first decoder in the  $l$ th iteration, that is, let  $\hat{q}_1^{(l)} \geq q_1^{(l)}(\mathbf{b}^i)$ , for  $i = 1, \dots, n$ . Similarly, let  $\hat{q}_2^{(l)}$  be an upper bound on the extrinsic information from the second decoder in the  $l$ th iteration. Under some appropriate conditions on the parameters  $\alpha_1, \alpha_2, \hat{p}_1$ , and  $\hat{p}_2$ , the following lemma provides a set of  $\hat{q}_1^{(l)}$  and  $\hat{q}_2^{(l)}$ .

*Lemma 2:* Assume that for a received vector  $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\mathbf{z}})$

- 1)  $\alpha_1 \alpha_2 < 1$
- 2)  $(\hat{p}_1 + \alpha_1 \hat{p}_2) \leq (1 - \alpha_1 \alpha_2)^2$
- 3)  $(\hat{p}_2 + \alpha_2 \hat{p}_1) \leq (1 - \alpha_1 \alpha_2)^2$ .

Under these conditions, starting from the unbiased initialization, the extrinsic information in the  $l$ th iteration for the  $i$ th information bit can be bounded as

$$\begin{aligned} q_1^{(l)}(\mathbf{b}^i) &< (\hat{p}_1 + \alpha_1 \hat{p}_2)(1 + (\alpha_1 \alpha_2) \\ &\quad + (\alpha_1 \alpha_2)^2 + \dots + (\alpha_1 \alpha_2)^{l-1}) + (\alpha_1 \alpha_2)^l \\ q_2^{(l)}(\mathbf{b}^i) &< (\hat{p}_2 + \alpha_2 \hat{p}_1)(1 + (\alpha_1 \alpha_2) \\ &\quad + (\alpha_1 \alpha_2)^2 + \dots + (\alpha_1 \alpha_2)^{l-1}) + (\alpha_1 \alpha_2)^l. \end{aligned}$$

*Proof:* The extrinsic information  $q_1^{(l)}(\mathbf{b}^i)$  is given by [see (9) and (10)]

$$\begin{aligned} q_1^{(l)}(\mathbf{b}^i) &= \frac{\sum_{\mathbf{b} \in \mathcal{H}_i} p_1(\mathbf{b}) p_0(\mathbf{b}) q_2^{(l-1)}(\mathbf{b})}{\sum_{\mathbf{b} \in \mathcal{H}_i^c} p_1(\mathbf{b}) p_0(\mathbf{b}) q_2^{(l-1)}(\mathbf{b})} \cdot \frac{1}{p_0(\mathbf{b}^i)} \cdot \frac{1}{q_2^{(l-1)}(\mathbf{b}^i)} \\ &= \frac{\sum_{\mathbf{b} \in \mathcal{H}_i} p_1(\mathbf{b}) \prod_{\substack{j: \mathbf{b} \in \mathcal{H}_j \\ j \neq i}} p_0(\mathbf{b}^j) q_2^{(l-1)}(\mathbf{b}^j)}{\sum_{\mathbf{b} \in \mathcal{H}_i^c} p_1(\mathbf{b}) \prod_{j: \mathbf{b} \in \mathcal{H}_j} p_0(\mathbf{b}^j) q_2^{(l-1)}(\mathbf{b}^j)}. \end{aligned} \quad (26)$$

Here, in order to obtain (26), we used the fact that  $p_0$  and  $q_2^{(l-1)}$  are product densities.

The denominator on the right-hand side of (26) is a sum of  $2^{n-1}$  positive terms corresponding to the  $2^{n-1}$  elements of  $\mathcal{H}_i^c$ . Regardless of the value of  $i$ , the set  $\mathcal{H}_i^c$  contains the all-zero sequence  $\mathbf{b}^0$ . Since the densities  $p_0$ ,  $p_1$ , and  $q_2^{(l-1)}$  are normalized, the term corresponding to  $\mathbf{b}^0$  equals 1, and the denominator on the right-hand side of (26) is greater than 1. It follows that

$$\begin{aligned} q_1^{(l)}(\mathbf{b}^i) &< \sum_{\mathbf{b} \in \mathcal{H}_i} p_1(\mathbf{b}) \prod_{\substack{j: \mathbf{b} \in \mathcal{H}_j \\ j \neq i}} p_0(\mathbf{b}^j) q_2^{(l-1)}(\mathbf{b}^j) \\ &\leq p_1(\mathbf{b}^i) + \hat{q}_2^{(l-1)} \sum_{\substack{\mathbf{b} \in \mathcal{H}_i \\ \mathbf{b} \neq \mathbf{b}^i}} p_1(\mathbf{b}) \prod_{\substack{j: \mathbf{b} \in \mathcal{H}_j \\ j \neq i}} p_0(\mathbf{b}^j) \end{aligned} \quad (27)$$

$$\leq p_1(\mathbf{b}^i) + \alpha_1 \hat{q}_2^{(l-1)}. \quad (28)$$

Here, in order to derive (27), we used the fact that  $\mathbf{b}^i$  is the only binary  $n$ -tuple in  $\mathcal{H}_i$  that, except for the  $i$ th bit, has no other nonzero bit, and that  $q_2^{(l-1)}(\mathbf{b}^j) \leq \hat{q}_2^{(l-1)} \leq 1$ . Later, it will be shown that indeed for the bounds proved here,  $\hat{q}_2^{(l-1)} \leq 1$ .

Since  $p_1(\mathbf{b}^i) \leq \hat{p}_1$ , we can now set the following upper bound on  $q_1^{(l)}(\mathbf{b}^i)$ :

$$\hat{q}_1^{(l)} = \hat{p}_1 + \alpha_1 \hat{q}_2^{(l-1)}. \quad (29)$$

A parallel argument for  $\hat{q}_2^{(l)}$  shows that

$$\hat{q}_2^{(l)} = \hat{p}_2 + \alpha_2 \hat{q}_1^{(l)}. \quad (30)$$

Combining (29) and (30), we get the following recursive relationships for  $\hat{q}_1^{(l)}$  and  $\hat{q}_2^{(l)}$ :

$$\hat{q}_1^{(l)} = \hat{p}_1 + \alpha_1 \hat{p}_2 + \alpha_1 \alpha_2 \hat{q}_1^{(l-1)} \quad (31)$$

$$\hat{q}_2^{(l)} = \hat{p}_2 + \alpha_2 \hat{p}_1 + \alpha_1 \alpha_2 \hat{q}_2^{(l-1)}. \quad (32)$$

Since the turbo decoding algorithm starts with the unbiased initialization, we can choose  $\hat{q}_1^{(0)} = \hat{q}_2^{(0)} = 1$ , and for  $l > 0$ , obtain

$$\begin{aligned} \hat{q}_1^{(l)} &= (\hat{p}_1 + \alpha_1 \hat{p}_2)(1 + (\alpha_1 \alpha_2) \\ &\quad + (\alpha_1 \alpha_2)^2 + \cdots + (\alpha_1 \alpha_2)^{l-1}) + (\alpha_1 \alpha_2)^l \end{aligned} \quad (33)$$

$$\begin{aligned} \hat{q}_2^{(l)} &= (\hat{p}_2 + \alpha_2 \hat{p}_1)(1 + (\alpha_1 \alpha_2) \\ &\quad + (\alpha_1 \alpha_2)^2 + \cdots + (\alpha_1 \alpha_2)^{l-1}) + (\alpha_1 \alpha_2)^l. \end{aligned} \quad (34)$$

In order to finish the proof, we need to show that the bounds given by (33) and (34) are indeed less than or equal to one. Given that  $\alpha_1 \alpha_2 < 1$  and  $(\hat{p}_1 + \alpha_1 \hat{p}_2) \leq (1 - \alpha_1 \alpha_2)^2$ , (33) implies that

$$\begin{aligned} \hat{q}_1^{(l)} &< \frac{\hat{p}_1 + \alpha_1 \hat{p}_2}{1 - \alpha_1 \alpha_2} + \alpha_1 \alpha_2 \\ &< 1. \end{aligned}$$

By a similar method, we can show that  $\hat{q}_2^{(l)} < 1$ .  $\square$

Next, we prove that the assumptions made in Lemma 2 can be satisfied with arbitrarily high probability by choosing the SNR high enough. To that end, we start by defining a *typical set*  $\mathfrak{Z} \subset \mathbb{R}^{n/r}$ ,  $n/r$  being the code length, such that if the received vector  $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\mathbf{z}})$  belongs to the set  $\mathfrak{Z}$ , then it satisfies the aforementioned assumptions. Let  $\mathfrak{Z}$  be the set of all received vectors that satisfy the following two conditions.

*Condition 1:* For  $i = 1, \dots, n$ ,  $p_0(\mathbf{b}^i) < \frac{1}{2^n}$ ,  $p_1(\mathbf{b}^i) < \frac{1}{4}$ , and  $p_2(\mathbf{b}^i) < \frac{1}{4}$ .

*Condition 2:* For  $\mathbf{b} \in \mathcal{H}$ ,  $p_1(\mathbf{b}) \leq 1$  and  $p_2(\mathbf{b}) \leq 1$ .

Let us verify that the received vectors in  $\mathfrak{Z}$  satisfy the assumptions made in Lemma 2. Clearly, for received vectors in  $\mathfrak{Z}$ ,  $\hat{p}_1 < 1/4$  and  $\hat{p}_2 < 1/4$ . Furthermore, the following calculation shows that for these vectors the parameter  $\alpha_1 < 1/2$ :

$$\begin{aligned} \alpha_1 &= \max_{i=1,2,\dots,n} \sum_{\substack{\mathbf{b} \in \mathcal{H}_i \\ \mathbf{b} \neq \mathbf{b}^i}} p_1(\mathbf{b}) \prod_{\substack{j: \mathbf{b} \in \mathcal{H}_j \\ j \neq i}} p_0(\mathbf{b}^j) \\ &< \max_{i=1,2,\dots,n} \sum_{\substack{\mathbf{b} \in \mathcal{H}_i \\ \mathbf{b} \neq \mathbf{b}^i}} \prod_{\substack{j: \mathbf{b} \in \mathcal{H}_j \\ j \neq i}} p_0(\mathbf{b}^j) < \frac{2^{n-1} - 1}{2^n} < \frac{1}{2}. \end{aligned} \quad (35)$$

Note that the summation in (35) is over  $2^{n-1} - 1$  terms with each term being less than  $1/2^n$ . By a similar calculation, it follows that  $\alpha_2 < 1/2$ . Now it is easy to verify that the received vectors in  $\mathfrak{Z}$  satisfy the assumptions made in Lemma 2.

The next lemma states that at asymptotically high SNRs, the received vector  $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\mathbf{z}})$  lies in the typical set  $\mathfrak{Z}$  (and hence satisfies the assumptions in Lemma 2) with high probability. The proof of this lemma is straightforward, but cumbersome. We relegate it to Appendix A.

*Lemma 3:* Given an  $\epsilon$  between 0 and 1, there exist a  $\sigma(\epsilon) > 0$  such that the probability of the received vector  $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\mathbf{z}})$  lying in the set  $\mathfrak{Z}$  is at least  $1 - \epsilon$  for  $0 < \sigma < \sigma(\epsilon)$ .

Finally, using Lemmas 2 and 3, we can show that at high SNRs, the turbo decoding algorithm has fixed points that correspond to the correct decisions on the information bits.

*Theorem 3:* Given any  $\epsilon$  between 0 and 1, we can choose a  $\sigma(\epsilon) > 0$  such that for  $0 < \sigma < \sigma(\epsilon)$ , the turbo decoding algorithm has, with probability more than  $1 - \epsilon$ , at least one fixed point that corresponds to correct decisions on the information bits. Moreover, starting from the unbiased initialization, the turbo decoding algorithm can converge to only one of these fixed points.

*Proof:* Throughout this proof, we will consider properties of the turbo decoding algorithm when the received vector

$(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\mathbf{z}})$  satisfies the conditions stipulated in Lemma 2. Now, assume that

$$q_j^{(l-1)}(\mathbf{b}^i) < 1, \quad j = 1 \text{ and } 2 \text{ and } i = 1, 2, \dots, n.$$

In this case, the upper bounds  $\hat{q}_1^{(l-1)}$  and  $\hat{q}_2^{(l-1)}$  can be chosen to be less than one. Now using (31) along with the conditions stipulated in Lemma 2, we obtain

$$\begin{aligned} \hat{q}_1^{(l)} &< \hat{p}_1 + \alpha_1 \hat{p}_2 + \alpha_1 \alpha_2 \\ &\leq (1 - \alpha_1 \alpha_2)^2 + \alpha_1 \alpha_2 \\ &< 1. \end{aligned} \quad (36)$$

Similarly, we can show that  $\hat{q}_2^{(l)} < 1$ . Therefore, for  $j=1$  and 2, and  $i=1, 2, \dots, n$ ,  $q_j^{(l-1)}(\mathbf{b}^i) < 1$  implies that  $q_j^{(l)}(\mathbf{b}^i) < 1$ .

The above calculation shows that the function iterated in the turbo decoding algorithm maps the unit-cube that extends from 0 to 1 in each dimension in  $\mathbb{R}^{2n}$ , to itself. By the Brouwer's fixed point theorem [5], the continuity of the iterated function implies that the decoding algorithm has a fixed point  $(q_1^*, q_2^*)$  such that  $q_j^*(\mathbf{b}^i) < 1$ , for  $j=1$  and 2,  $i=1, 2, \dots, n$ . Clearly, this fixed point corresponds to correct decisions on the information bits.

Finally, note that by choosing the SNR high enough, the probability of the received vector satisfying the conditions stipulated in Lemma 2, can be made arbitrarily close to 1.  $\square$

Theorem 3 is somewhat unsatisfactory since it does not say if such fixed points are unique. Furthermore, just as in the low SNR case, we have no guarantee that the turbo decoding algorithm will converge to one of the fixed points shown here. The next section provides a partial answer for these questions. Simulations show that for moderate SNRs, the turbo decoding algorithm does converge to a fixed point that corresponds to the correct decisions. Still, the simulation results do not resolve the question of the uniqueness.

## V. FIXED POINTS AT PRACTICAL SNRS

The existence of certain special fixed points at *asymptotic* SNRs raises interesting questions. Does the turbo decoding algorithm with unbiased initialization converge to these fixed points? If so, what are the threshold values of SNR beyond which the turbo decoding algorithm converges to these fixed points? It is also interesting to investigate relationships between the fixed points at practical SNRs and those at asymptotic SNRs.

To answer these questions, we first performed extensive simulations. In our simulations, we used the classical turbo codes with identical constituent recursive convolutional codes generated by the polynomials  $\{D^4 + D^3 + D^2 + D + 1, D^4 + 1\}$ . Parity bits were punctured alternately to produce a rate-1/2 turbo code. The codewords were transmitted over an AWGN channel using the BPSK modulation.

Simulation results show that for a fixed interleaver length  $n$ , there is a threshold value  $S_l$  such that if the SNR is *less* than  $S_l$ , then the decoding algorithm converges in all *simulated* cases. Similarly, there is another threshold  $S_h$  such that the decoding algorithm converges in all *simulated* cases for SNRs *more* than  $S_h$ . Fig. 1 shows these thresholds for classical turbo codes with

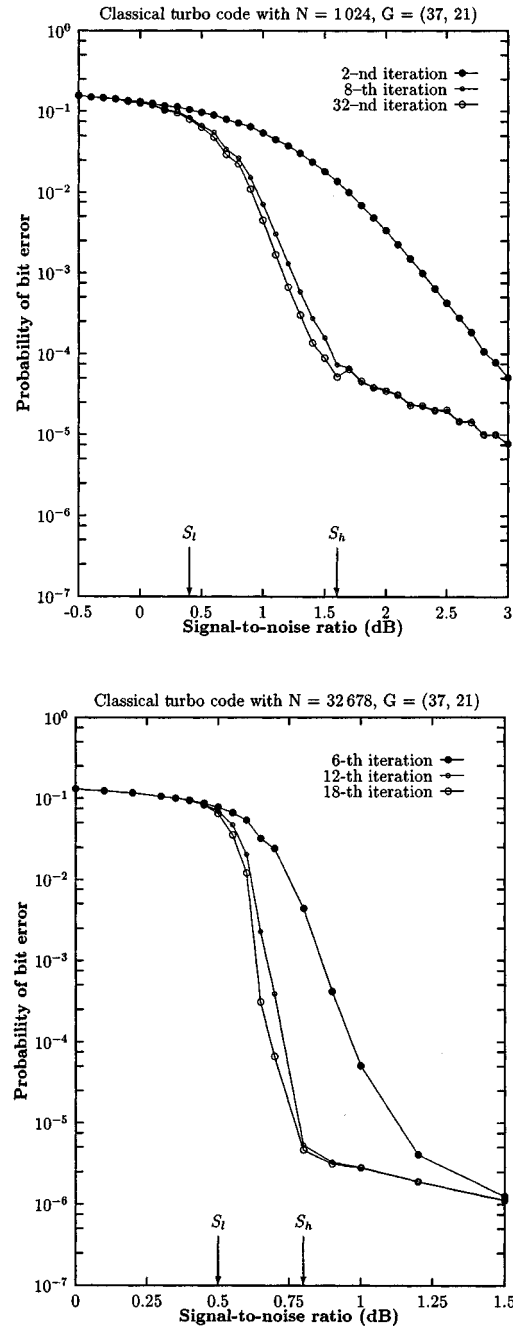


Fig. 1. The performance of classical turbo codes with interleaver length 1024 and 32 768.  $S_l$  and  $S_h$  are the SNR thresholds beyond which the turbo decoding algorithm converges and exhibits asymptotic behavior.

interleaver lengths  $2^{10} = 1024$  and  $2^{15} = 32768$ . Similar thresholds can also be observed for interleaver lengths between  $2^{10}$  to  $2^{15}$ .

Note that these thresholds are defined by the convergence of the turbo decoding algorithm. We obtained these thresholds by simulating the turbo decoding of a large number of received vectors for each SNR point. It is the relationship of the threshold values to the performance curve, not the exact threshold values, that is of interest here. We observe that for all interleaver lengths, the performance of turbo codes improves sharply as the SNR changes from  $S_l$  to  $S_h$  (see Fig. 1).

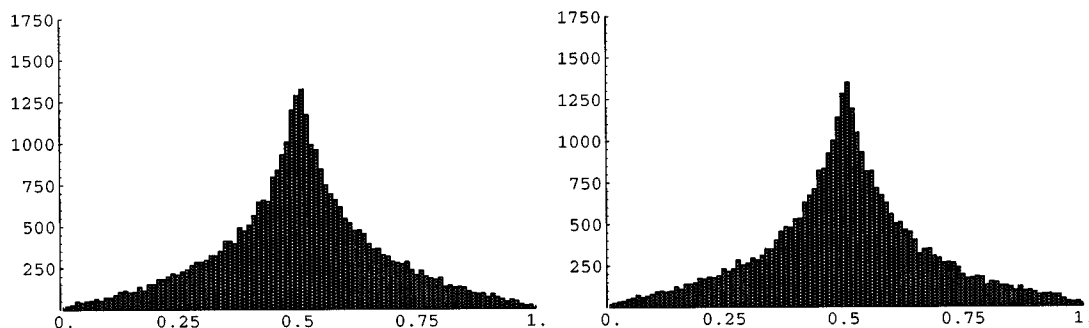


Fig. 2.  $\Pr_{q_1^*}(b_i = 0)$  and  $\Pr_{q_2^*}(b_i = 0)$ , for  $i = 1, \dots, n$ , at SNR = 0.0 dB in a histogram representation. Note that the histograms are weighed heavily around 0.5 indicating the indecisive nature of the extrinsic information.

For SNRs between  $S_l$  and  $S_h$ , the turbo decoding algorithm *may* not converge even after several thousand iterations. In contrast to this behavior, for SNRs outside the range  $(S_l, S_h)$ , the turbo decoding algorithm converges rapidly—typically, in 10 to 30 iterations. In a few cases, the convergence may take longer and may need a few hundred iterations. As the SNR becomes progressively larger than  $S_h$  (or lower than  $S_l$ ), the cases of slow convergence become less frequent, and ultimately for high enough (or low enough) SNRs, the convergence becomes strong and rapid in all simulated cases.

In the rest of this section, we examine the characteristics of fixed points at practical SNRs, and compare them with the corresponding characteristics of fixed points at asymptotic SNRs.

#### A. Characteristics of Fixed Points at Practical SNRs

Before proceeding further, it is helpful to consider Theorem 1 when the log-densities  $P_1$  and  $P_2$  are product log-densities. In that case, the equations iterated in the turbo decoding algorithm can be simplified as

$$\begin{aligned} Q_1 &\leftarrow \pi_{P_1}(P_0 + Q_2) - (P_0 + Q_2) \\ Q_1 &\leftarrow P_1 \end{aligned} \quad (37)$$

and

$$\begin{aligned} Q_2 &\leftarrow \pi_{P_2}(P_0 + Q_1) - (P_0 + Q_1) \\ Q_2 &\leftarrow P_2. \end{aligned} \quad (38)$$

Here, we used the fact that  $\pi_P(Q)$  is the *unique* (normalized) product log-density that has the same bitwise marginals as  $P + Q$ . Hence, if  $P$  is a product log-density, then  $\pi_P(Q)$  equals  $P + Q$ . Thus, if  $P_1$  and  $P_2$  are product log-densities, the turbo decoding algorithm converges to  $(Q_1^*, Q_2^*) = (P_1, P_2)$  in a single iteration regardless of its initialization.

The continuity of the fixed points with respect to the parameters  $(P_1, P_2)$  implies that if  $(P_1, P_2)$  is close enough to a pair of product log-densities  $(\hat{P}_1, \hat{P}_2)$ , then the turbo decoding algorithm has a unique fixed point close to  $(\hat{P}_1, \hat{P}_2)$ . Moreover, we expect that the unbiased initialization  $(\mathbf{0}, \mathbf{0})$  will be in the domain of attraction, since for product densities, the domain of attraction is  $\Pi \times \Pi$  which includes the unbiased initialization.

As shown in the previous section, for asymptotically low SNRs,  $P_1$  and  $P_2$  converge to the product log-density  $\mathbf{0}$ . Therefore, at asymptotically low SNRs, the turbo decoding

algorithm should have a fixed point close to  $(\mathbf{0}, \mathbf{0})$ . Simulations show that not only is this true but the SNR required for the existence of this fixed point is not extremely low. In fact, for SNRs below  $S_l$ , the turbo decoding algorithm converges and most of the extrinsic log-likelihood ratios,  $Q_1^*(b^i)$  and  $Q_2^*(b^i)$ , for  $i = 1, \dots, n$ , are close to 0.0. We refer to a fixed point with these characteristics as an *indecisive* fixed point because at such fixed points, the turbo decoding algorithm is relatively ambiguous regarding the values of the information bits.

Fig. 2 shows the characteristics of a fixed point  $(Q_1^*, Q_2^*)$  obtained by simulating the turbo decoding algorithm at 0.0 dB with an interleaver of length 32 768. In order to obtain this fixed point (and all other fixed points discussed in this section), we performed enough iterations so that, within six significant digits,  $(Q_1, Q_2)$  did not change in successive iterations. Fig. 2 plots  $\Pr_{q_1^*}(b_i = 0)$  and  $\Pr_{q_2^*}(b_i = 0)$  for  $i = 1, \dots, n$ , in a histogram format, where  $\Pr_{q_1^*}$  and  $\Pr_{q_2^*}$  are the probability measures induced by  $Q_1^*$  and  $Q_2^*$ , respectively. These histograms show that  $\Pr_{q_1^*}(b_i = 0)$  and  $\Pr_{q_2^*}(b_i = 0)$  are clustered around 0.5. This corroborates the inferences made above by extrapolating the characteristics of fixed points at asymptotically low SNRs.

Fig. 3 presents the same statistics for a fixed point at a higher SNR of 0.35 dB. A comparison of Figs. 2 and 3 shows that the histograms for 0.0 dB are weighted more around 0.5 than the histograms for 0.35 dB. This is a general trend: as the SNR becomes lower, the fixed point  $(Q_1^*, Q_2^*)$  gets closer to the point  $(\mathbf{0}, \mathbf{0})$ .

On the other hand, simulations at high SNRs show that in most cases, the log-densities  $Q_1^*$  and  $Q_2^*$  are concentrated on the information sequence that corresponds to the codeword closest to the received vector. In other words, with high probability,  $\Pr_{q_1^*}(b_i = 0)$  and  $\Pr_{q_2^*}(b_i = 0)$  are either approximately 0 or approximately 1, depending upon whether the  $i$ th information bit of the codeword closest to the received vector is 1 or 0 (see Fig. 4). Since  $\Pr_{q_1^*}(b_i = 0)$  and  $\Pr_{q_2^*}(b_i = 0)$  are either close to 0 or close to 1, the final log-likelihood ratios computed by the decoding algorithm are strong indications of the information bits, and we refer to a fixed point with these characteristics as an *unequivocal* fixed point.

The characteristics of an unequivocal fixed point are similar to the characteristics predicted for fixed points at asymptotically large SNRs. At large SNRs, given that the all zero codeword was

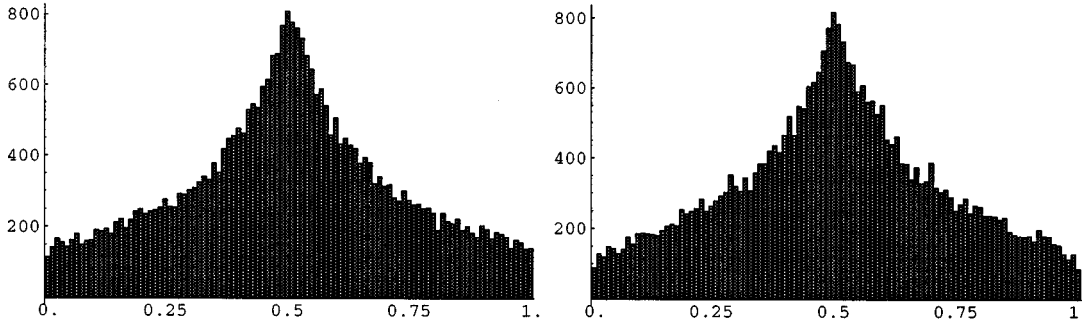


Fig. 3.  $\Pr_{q_1^*}(b_i = 0)$  and  $\Pr_{q_2^*}(b_i = 0)$ , for  $i = 1, \dots, n$ , at SNR = 0.35 dB in a histogram representation. These histograms have less weight around 0.5 compared to the histograms at 0.0 dB.

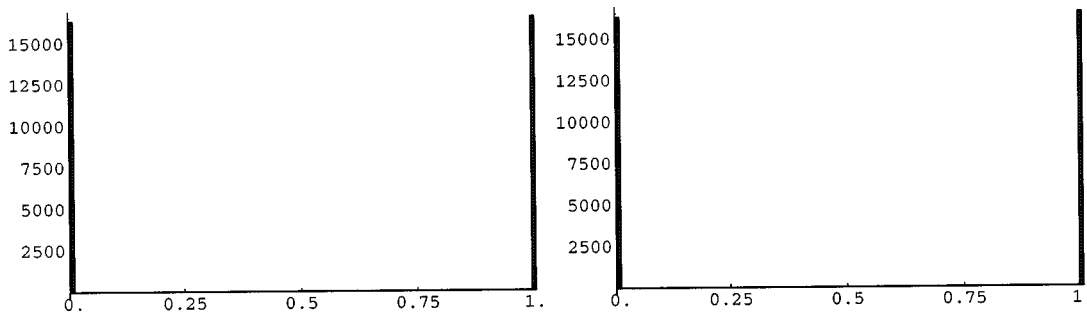


Fig. 4. At SNR = 0.80 dB,  $\Pr_{q_1^*}(b_i = 0)$  and  $\Pr_{q_2^*}(b_i = 0)$ , for  $i = 1, \dots, n$ , are either close to 0 or close to 1. Similar histograms are obtained for SNRs greater than  $S_h$ .

transmitted, the conditions stipulated in Lemma 2 are satisfied with high probability and  $\alpha_1 \alpha_2 \ll \hat{p}_1 + \alpha_1 \hat{p}_2 < 1$ . Hence, the extrinsic information  $q_1(\mathbf{b}^i)$  from the first decoder can be upper-bounded approximately by  $\hat{p}_1 + \alpha_1 \hat{p}_2$  which, in turn, is of the order of  $\exp(-4/\sigma^2)$  [cf. (19)]. Here we assume that an information sequence of weight one generates a parity sequence of weight at least two from each constituent code; an assumption satisfied by most turbo codes. Therefore, at large SNRs, for a fixed point  $(q_1^*, q_2^*)$ , shown to exist in Section IV-B

$$q_1^*(\mathbf{b}^i) < \exp(-4/\sigma^2)$$

and

$$q_2^*(\mathbf{b}^i) < \exp(-4/\sigma^2)$$

or, in other words, the probability measures  $\Pr_{q_1^*}$  and  $\Pr_{q_2^*}$  are concentrated on the transmitted codeword.

Simulation results reveal that the turbo decoding algorithm needs only a moderate SNR to converge to an unequivocal fixed point. For example, Fig. 4 shows  $\Pr_{q_1^*}(b_i = 0)$  and  $\Pr_{q_2^*}(b_i = 0)$  for  $i = 1, \dots, n$ , in a histogram for an unequivocal fixed point at SNR = 0.8 dB. In this case, instead of the all-zero codeword, a random codeword of length 32 768 was transmitted. As a result, for about 16 384 information bits, extrinsic information is close to 0, and for others, it is close to 1. In general, we observe a similar trend: for SNRs higher than  $S_h$ , the turbo decoding algorithm converges to an unequivocal fixed point.

Apart from the differences in the distribution of the final likelihood ratios, there is another important distinction between

these two types of fixed points. In all simulated cases, hard decisions corresponding to an unequivocal fixed point formed a codeword, while for an indecisive fixed point, hard decisions did not form a codeword.

In the intermediate range, for SNRs between  $S_l$  and  $S_h$ , the turbo decoding algorithm may not converge. In case the decoding algorithm converges, depending on the parameters  $P_0$ ,  $P_1$ , and  $P_2$ , it may converge either to an unequivocal fixed point or to an indecisive fixed point. For an indecisive fixed point, the extrinsic information is further away from  $\mathbf{0}$  than it is for SNRs less than  $S_l$  (see Fig. 5). In contrast, for an unequivocal fixed point, the extrinsic information is the same as for a fixed point at high SNR (see Fig. 6).

An interesting behavior in the intermediate range is the quasi-periodic phase trajectory of the turbo decoding algorithm. For some values of  $P_0$ ,  $P_1$ , and  $P_2$ , the phase trajectory of the turbo decoding algorithm becomes quasi-periodic after a transient period. That is, the phases

$$(Q_1^{(k)}, Q_2^{(k)}), (Q_1^{(k+1)}, Q_2^{(k+1)}), \dots$$

are very close to the phases

$$(Q_1^{(k-u)}, Q_2^{(k-u)}), (Q_1^{(k-u+1)}, Q_2^{(k-u+1)}), \dots$$

respectively, where  $u$  is the “period” of the phase trajectory. In a few cases, the phase trajectory may even become periodic. The extrinsic information in a quasi-periodic trajectory is somewhere in between the extrinsic information for an indecisive and

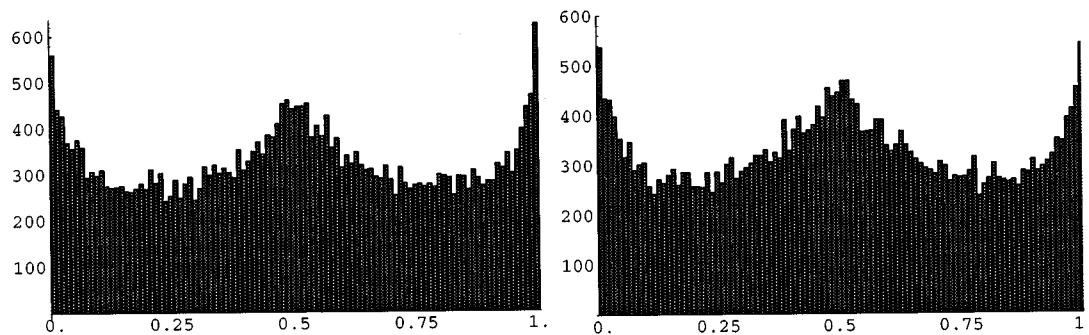


Fig. 5. The histograms of  $\Pr_{q_1^*}(b_i = 0)$  and  $\Pr_{q_2^*}(b_i = 0)$  at 0.60 dB. In this case, the fixed point is an indecisive fixed point.

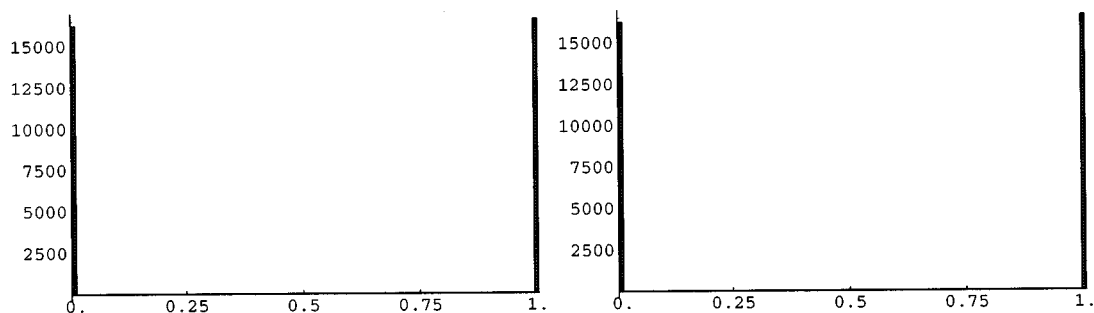


Fig. 6. The histograms of  $\Pr_{q_1^*}(b_i = 0)$  and  $\Pr_{q_2^*}(b_i = 0)$  at 0.60 dB. In this case, the fixed point is an unequivocal fixed point.

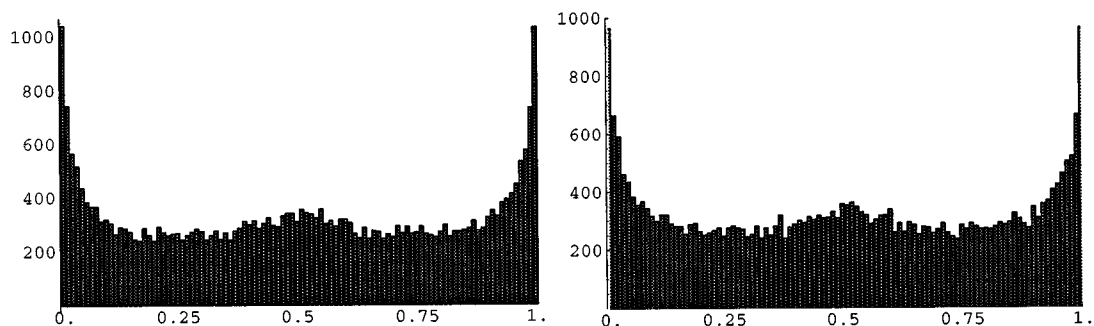


Fig. 7. The histograms of  $\Pr_{q_1}(b_i = 0)$  and  $\Pr_{q_2}(b_i = 0)$  for a randomly chosen phase in a quasi-periodic phase trajectory at 0.60 dB.

an unequivocal fixed point. Fig. 7 shows one such case at 0.6-dB SNR for a classical turbo code with interleaver length 32 678.

## VI. CONTINUATION OF FIXED POINTS WITH SNR

Simulation results provided in the preceding section raise some interesting questions. What happens to unequivocal fixed points as the SNR is reduced beyond the waterfall region? Are the quasi-periodic phase trajectories of the turbo decoding algorithm related to its fixed points? To answer such questions, we treated the turbo decoding algorithm as a dynamical system parameterized approximately by the SNR and studied behavior of its phase trajectories as the SNR is varied.

In this section, we show that, in fact, both types of fixed points coexist for SNRs well below  $S_l$ . However, at low SNRs after the unbiased initialization, the turbo decoding algorithm converges

to an indecisive fixed point. We show that as the SNR increases, indecisive fixed points become less and less stable, and finally they bifurcate in the waterfall region between  $S_l$  and  $S_h$ . The bifurcation of indecisive fixed points enables the turbo decoding algorithm to converge to an unequivocal fixed point. The bifurcation of indecisive fixed points also explains the quasi-periodic and periodic phase trajectories observed in simulations for the SNRs in the waterfall region.

### A. The Turbo Decoding Algorithm as a Single-Parameter Dynamical System

As a discrete dynamical system, the turbo decoding algorithm is parameterized by the log-densities  $P_0$ ,  $P_1$ , and  $P_2$ . Given the transmitted codeword,  $P_0$ ,  $P_1$ , and  $P_2$  are completely specified by the noise values  $\nu_1, \nu_2, \dots, \nu_{2n}$ , where we assume that a rate-1/2 code with code length  $2n$  is used. It follows that we

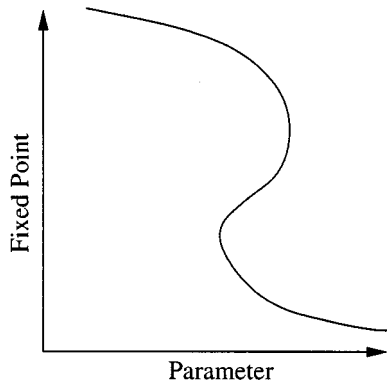


Fig. 8. Implicit function theorem implies that a fixed point moves smoothly with the system parameter.

can view the turbo decoding algorithm as being parameterized by  $\nu_1, \nu_2, \dots, \nu_{2n}$ .

To study the phase trajectories of the turbo decoding algorithm, we would like to parameterize it by the SNR  $1/\sigma^2$ . Fortunately, the SNR has a good approximation in terms of  $\nu_1, \nu_2, \dots, \nu_{2n}$ . For a large codeword length, typical of turbo codes, the noise variance  $\sigma^2$  is approximately equal to

$$\hat{\sigma}^2 = \sum_{i=1}^{2n} \nu_i^2 / 2n.$$

Therefore, we fix  $(2n - 1)$  noise ratios  $\nu_1/\nu_2, \nu_2/\nu_3, \dots, \nu_{2n-1}/\nu_{2n}$  and treat the turbo decoding algorithm as being parameterized by  $\hat{\sigma}^2$  alone. We used a large number of randomly chosen sets of noise ratios to get an understanding of the typical behavior of the turbo decoding algorithm. Since bifurcation theory is well developed for a single parameter system, the above parameterization also helps in applying the bifurcation theory to the analysis of phase trajectories. For some background on bifurcation theory, we refer the reader to [9], [16].

We use the *continuation principle* to study variations of stable fixed points with respect to  $\hat{\sigma}^2$ . The continuation principle is based on the Implicit Function Theorem which states that if for  $\hat{\sigma}^2 = \mu_0$ ,  $(Q_1^*(\mu_0), Q_2^*(\mu_0))$  is a fixed point of turbo decoding, then under certain mild conditions, for the values of  $\hat{\sigma}^2$  in the vicinity of  $\mu_0$ , the turbo decoding algorithm has a unique fixed point close to  $(Q_1^*(\mu_0), Q_2^*(\mu_0))$ . In other words, under certain sufficient conditions, a small change in  $\hat{\sigma}^2$  produces a small change in the fixed point.

For turbo decoding, these conditions are satisfied by a *stable* fixed point. In order to find the change in a stable fixed point, when  $\hat{\sigma}^2$  changes by a small amount from  $\mu_0$  to  $\mu_1$ , the continuation principle uses  $(Q_1^*(\mu_0), Q_2^*(\mu_0))$  to initialize the turbo decoding algorithm. Since  $(Q_1^*(\mu_0), Q_2^*(\mu_0))$  is close to the stable fixed point for  $\hat{\sigma}^2 = \mu_1$ , the iterations converge quickly to  $(Q_1^*(\mu_1), Q_2^*(\mu_1))$ . Next,  $(Q_1^*(\mu_1), Q_2^*(\mu_1))$  is used for initialization to find  $(Q_1^*(\mu_2), Q_2^*(\mu_2))$ , where  $\mu_2$  is sufficiently close to  $\mu_1$ , and so on. This produces what is known as the *equilibrium curve* of a fixed point for a range of  $\hat{\sigma}^2$  (see Fig. 8).

### B. Continuation of Unequivocal Fixed Points

To obtain the equilibrium curve of an unequivocal fixed point, we start from a realization of noise values that corresponds to a very high SNR (bit-error rate less than  $10^{-10}$ ) and run the turbo decoding algorithm until it converges. Next, as outlined earlier, we change noise values  $\nu_1, \dots, \nu_{2n}$  in such a way that  $\hat{\sigma}^2 = \sum_{i=1}^{2n} \nu_i^2 / 2n$  increases by a small amount while the  $(2n - 1)$  ratios of  $\nu_1, \dots, \nu_{2n}$  remain constant. The turbo decoding algorithm is run again, initialized with the previous fixed point, to find the fixed point corresponding to the new noise values. This process is repeated until  $\hat{\sigma}^2$  corresponds to SNRs well below  $S_t$ .

Extensive simulations, using a large number of random noise-samples, show that in this wide range of SNRs, from the SNR corresponding to the bit-error rate of  $10^{-10}$  to the SNRs well below  $S_t$ , an unequivocal fixed point barely changes. The distribution of extrinsic information for this range remains similar to that shown in Fig. 4, and with high probability, hard decisions implied by the extrinsic information correspond to the correct decisions on the information bits.

These experiments demonstrate that for SNRs well below  $S_t$ , the turbo decoding algorithm possesses multiple fixed points, including an unequivocal fixed point. However, unequivocal fixed points are “shadowed” by indecisive fixed points, and the turbo decoding algorithm starting from the unbiased initialization does not reach an unequivocal fixed point at low SNRs.

### C. Continuation of Indecisive Fixed Points

To obtain an equilibrium curve of an indecisive fixed point, we start from a realization of noise values that corresponds to a very low SNR (bit-error rate more than  $10^{-1}$ ), and run the turbo decoding algorithm until it converges to an indecisive fixed point. Afterwards, we decrease  $\hat{\sigma}^2 = \sum_{i=1}^{2n} \nu_i^2 / 2n$  by a small amount, while keeping the  $(2n - 1)$  ratios of  $\nu_1, \dots, \nu_{2n}$  constant. The turbo decoding algorithm is run again, initialized with the previous fixed point, to find the fixed point corresponding to the new noise values.

Unlike the unequivocal fixed points, an indecisive fixed point varies significantly with  $\hat{\sigma}^2$ . Consequently, its domain of attraction as well as its stability changes significantly. In all cases that we tried, indecisive fixed points lose their stability in the waterfall region.

Recall that a fixed point is stable if and only if the Jacobian of the iterated function, evaluated at the fixed point, has all its eigenvalues inside the unit circle. When a fixed point (and along with it, the eigenvalues evaluated at the fixed point) varies continuously with  $\hat{\sigma}^2$ , it can *bifurcate* (see Appendix C) and lose its stability by one of the following three mechanisms.

- A pair of complex conjugate eigenvalues approach the unit circle. This corresponds to the *Neimark–Sacker bifurcation* of the fixed point (see Fig. 9)
- An eigenvalue approaches  $-1$ . This corresponds to the *flip bifurcation* of the fixed point.
- An eigenvalue approaches  $+1$ . This corresponds to the *fold bifurcation* of the fixed point.

In our simulation experiments, we found that all three mechanisms were responsible for the bifurcation of indecisive fixed

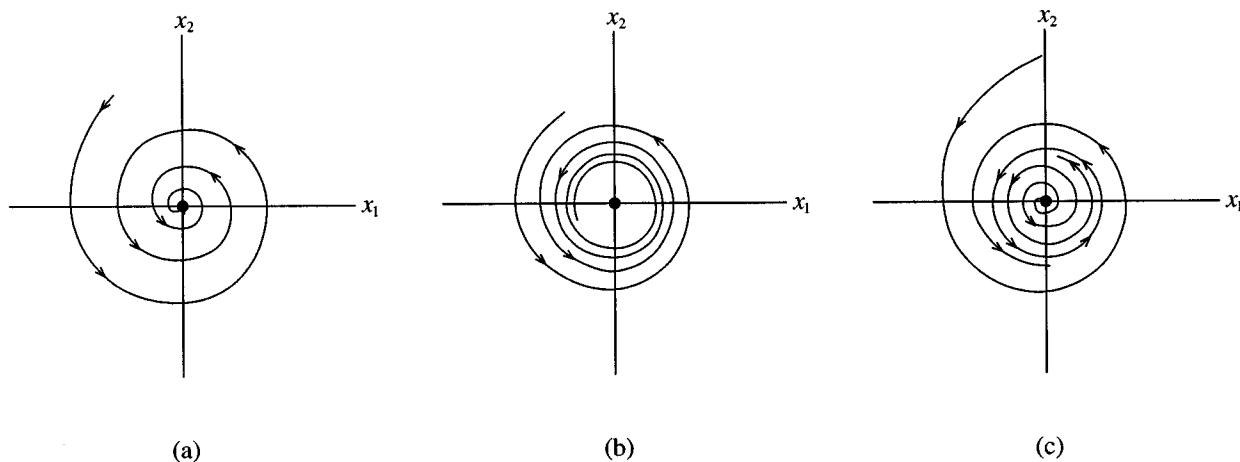


Fig. 9. This figure shows typical changes in phase trajectories induced by the Neimark–Sacker bifurcation of a fixed point in a two-dimensional system. Here, the stable fixed point is at the origin. (a) Before bifurcation, the fixed point is stable and phase trajectories starting in its neighborhood converge to it. (b) When a pair of eigenvalues reaches the unit circle, the fixed point remains (nonlinearly) stable. As a result, phase trajectories still converge to the fixed point, although they take much longer to converge. (c) After the Neimark–Sacker bifurcation, the fixed point is surrounded by an invariant and stable curve. Now, the phase trajectories starting in the neighborhood of the fixed point converge to this invariant curve.

points. Each of these mechanisms results in a phase trajectory that has distinguishing characteristics. In the rest of this section, we discuss these cases in more detail.

Before proceeding further, we note that the Jacobian is computed by using the alternative formulation of the turbo decoding algorithm given in Appendix B. This formulation uses  $P_t = P_0 + Q_1 + Q_2$  as the iterated variable. Since the phase  $P_t$  is a multidimensional variable, it is not possible to plot the entire phase trajectory on the two dimensions of this paper. Instead, in Figs. 10–25, the  $y$ -axis represents the sum  $\sum_{i=1}^n \text{Pr}_{P_t^*}^2(b_i = 0)$ . This sum is the squared sum of the posterior probabilities of the information bits being equal to 0. The  $x$ -axis denotes the iteration number. Furthermore, the phase trajectories, given in Figs. 10–25, correspond to the transmission of the all-zero codeword of a classical turbo code with interleaver length 1024. Therefore, the proximity of the sum  $\sum_{i=1}^n \text{Pr}_{P_t^*}^2(b_i = 0)$  to 1024 indicates the proximity of  $P_t^*$  to an unequivocal fixed point with correct decisions on the information bits. In particular,  $\sum_{i=1}^n \text{Pr}_{P_t^*}^2(b_i = 0) \approx 1024$  indicates the convergence to an unequivocal fixed point.

1) *Neimark–Sacker Bifurcation of Indecisive Fixed Points:* In a two-dimensional system (a system with two variables), the Neimark–Sacker bifurcation of a stable fixed point makes the fixed point unstable. After the bifurcation, the resulting unstable fixed point is surrounded by an isolated invariant curve that is unique, closed, and asymptotically stable. An invariant set is a set of phases such that once a phase trajectory enters the invariant set, it never leaves that set. Initially, right after the bifurcation, the invariant curve resembles a circle (see Fig. 9). The phase trajectories in this approximately circular invariant curve are either periodic or quasi-periodic [16]. As the system parameter changes further, the invariant curve may become irregular and lose its stability.

In  $n$ -dimensional turbo decoding algorithm, by the Reduction Principle (see Appendix C), modes corresponding to eigenvalues inside the unit circle decay exponentially fast in the neighborhood of a stable fixed point undergoing the Neimark–

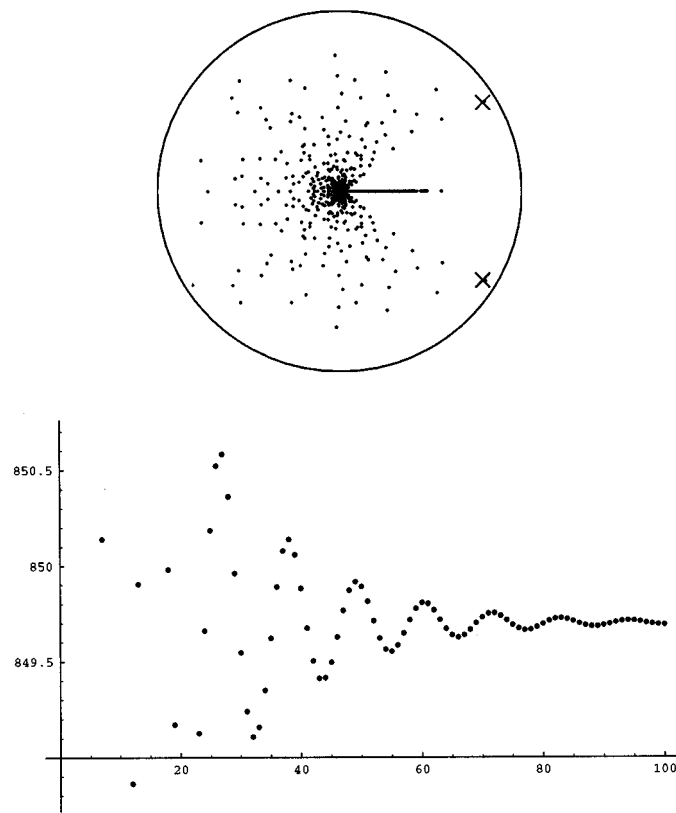


Fig. 10. At SNR = 0.65 dB, the turbo decoding algorithm converges to a stable indecisive fixed point. The eigenvalues evaluated at this fixed point lie inside the unit circle, although a pair of complex conjugate eigenvalues is close to the unit circle.

Sacker bifurcation. As a result, after a transient period, phase trajectories in a neighborhood of a stable fixed point undergoing the Neimark–Sacker bifurcation are well approximated by the corresponding phase trajectories in a two-dimensional system.

Figs. 10–15 illustrate the Neimark–Sacker bifurcation of a stable fixed point for the turbo decoding algorithm. As expected,

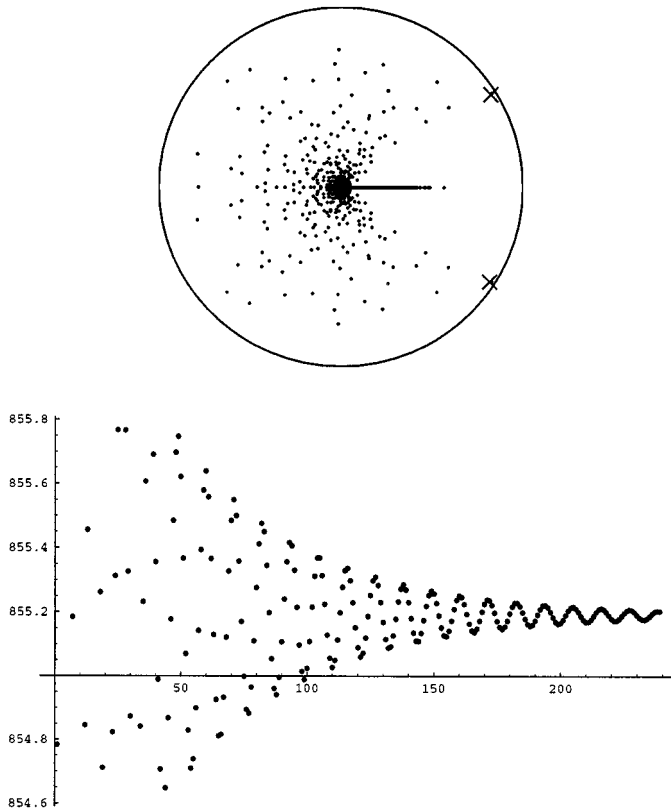


Fig. 11. As the SNR is increased to 0.67 dB, the pair of eigenvalues moves closer to the unit circle and the turbo decoding algorithm takes longer to converge.

just before the bifurcation, at SNR = 0.65 and 0.67 dB, the phase trajectory spirals down and converges to a stable indecisive fixed point. At SNR = 0.68 dB, this fixed point has undergone the Neimark–Sacker bifurcation. As a result, after a transient period, the phase trajectory approaches the resulting asymptotically stable invariant curve and becomes quasi-periodic. Initially, when the system parameter is close to the critical value, the invariant curve is nearly circular and stable. As the SNR increases, the invariant curve becomes more irregular and finally loses its stability at SNR = 0.85 dB. As soon as the invariant curve loses its stability, the turbo decoding algorithm is able to find an unequivocal fixed point corresponding to the transmitted codeword (Fig. 15).

2) *Flip Bifurcation of Indecisive Fixed Points:* In contrast to the Neimark–Sacker bifurcation, a flip bifurcation can take place in a one-dimensional system. Fig. 16 shows the phase trajectories of a typical one-dimensional system undergoing a flip bifurcation. Note that as a result of flip bifurcation, a stable fixed point becomes unstable, and an asymptotically stable cycle of period two appears in the neighborhood of the resulting unstable fixed point. Initial conditions that before the bifurcation resulted in a phase trajectory converging to the stable fixed point are now likely to result in a phase trajectory that converges to the cycle of period two (Fig. 16). As the system parameter moves further away from its critical value, this cycle may either lose its stability or bifurcate into a cycle of period four [9].

By the Reduction Principle, we expect a similar behavior from the turbo decoding algorithm.

Figs. 17–21 show typical phase trajectories that start in the neighborhood of an indecisive fixed point before and after the flip bifurcation. In this particular case, as the SNR increases, a stable indecisive fixed point bifurcates and the phase trajectories starting in the neighborhood of the fixed point settle in a cycle of period two. On increasing the SNR further to 0.70 dB, the cycle becomes unstable, and as soon as that happens, the turbo decoding algorithm finds an unequivocal fixed point that corresponds to the transmitted codeword.

3) *Fold Bifurcation of Indecisive Fixed Points:* Fig. 22 shows changes in the phase trajectories induced by a typical fold bifurcation of a stable fixed point in a one-dimensional dynamical system. In contrast to Neimark–Sacker and flip bifurcations, a fixed point disappears after fold bifurcation. Moreover, the fold bifurcation of a fixed point does not result in an invariant set in its neighborhood.

Again, by the Reduction Principle, we expect similar phase trajectories in the turbo decoding algorithm. Figs. 23–25 show phase trajectories starting in the neighborhood of a fixed point that undergoes a fold bifurcation. Initially at SNR 0.72 dB, phase trajectories converge to the fixed point. As the SNR is increased to 0.7244 dB (see Fig. 24), the fixed point becomes less stable and phase trajectories take longer to converge. Finally, at 0.73-dB SNR, the fixed point disappears due to a fold bifurcation. Since the fold bifurcation does not result in an invariant set, phase trajectories are able to move away from this neighborhood and are able to find an unequivocal fixed point. Notice that even a small change in SNR induces an abrupt change in the fixed point and, as a consequence, an abrupt change in the nature of phase trajectories.

In summary, the continuation of unequivocal fixed points shows that both indecisive and unequivocal fixed points coexist for SNRs well below  $S_l$ . As the SNR increases, indecisive fixed points bifurcate in the waterfall region and often result in stable invariant sets in their neighborhoods. These invariant sets prevent the turbo decoding algorithm from converging to an unequivocal fixed point. However, as the SNR further increases to  $S_h$ , the invariant sets also become unstable or disappear. As soon as this happens, the turbo decoding algorithm is able to converge an unequivocal fixed point.

## VII. A NEW STOPPING CRITERION FOR TURBO DECODING

As we have seen in the previous sections, typically, a phase trajectory of the turbo decoding algorithm has one of the following three asymptotic behaviors: convergence to an unequivocal fixed point, convergence to an indecisive fixed point, or reaching an invariant set. While a phase trajectory converging to an unequivocal fixed point results in zero or a small number of bit errors, other phase trajectories correspond to a relatively large number of bit errors. In this section, we propose a new stopping criterion which exploits the transient behavior of a phase trajectory to predict its asymptotic behavior, and, as a corollary, which predicts the tradeoff between efforts spent in continuing more iterations and the resulting reduction in bit errors.

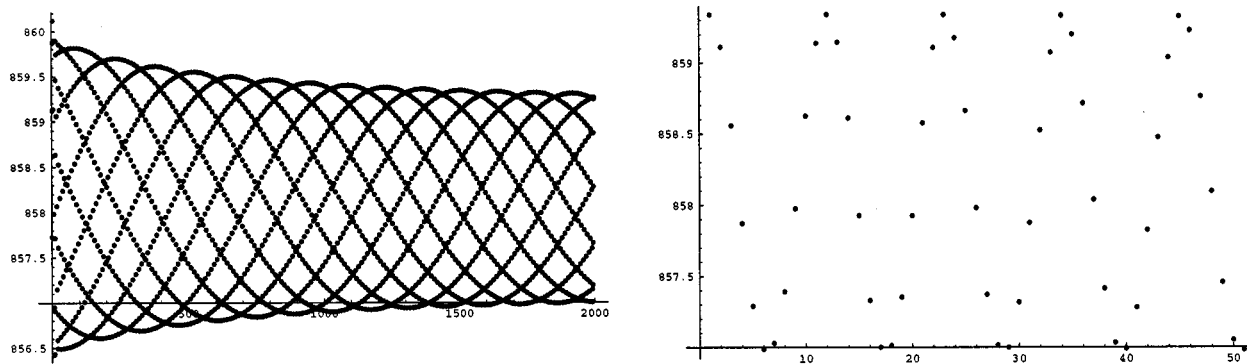


Fig. 12. On increasing the SNR further to 0.68 dB, the stable fixed point undergoes a Neimark–Sacker bifurcation. After a transient period, the phase trajectory, shown on the left-hand side, becomes quasi-periodic. The figure on the right-hand side shows a magnified view of the phase trajectory from the iteration number 1000 to the iteration number 1050. Note that the trajectory is not quite periodic, instead it is quasi-periodic.

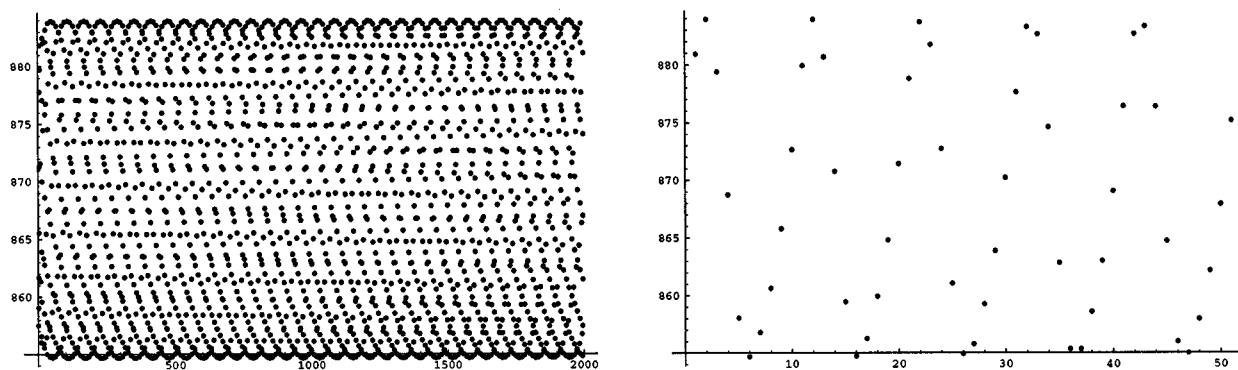


Fig. 13. As the SNR increases, the invariant curve becomes less circular and less stable. The phase trajectory shown here corresponds to SNR = 0.71 dB.

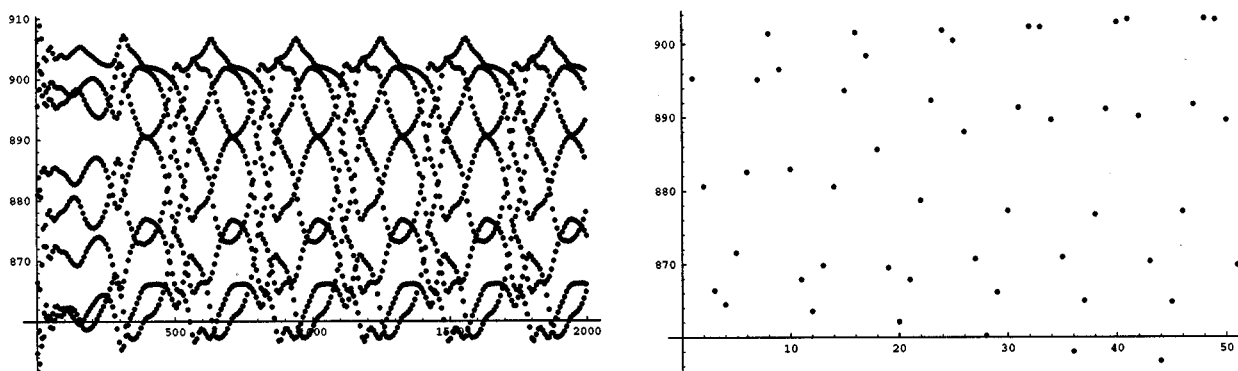


Fig. 14. On increasing the SNR further to 0.75 dB, the invariant curve becomes more irregular. The magnified trajectory on the right shows that corresponding phases in successive periods are less correlated.

The proposed stopping criterion is not only simpler to implement, but it also has a better performance than other previously known stopping criteria, such as the cross-entropy (CE) criterion [8], and the variants thereof [14]. The stopping criterion given here is just a representative of stopping criteria that can

be designed based on the similar principles. In practice, the design of a stopping criterion will take into account the range of SNRs under consideration, tolerance of the intended application to the suboptimality introduced by an aggressive stopping criterion, etc. Such fine tuning is outside the scope of this work.

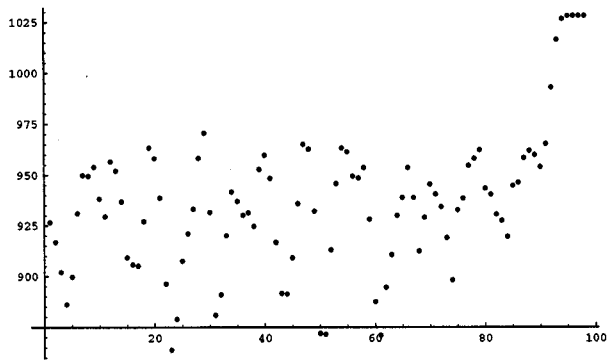


Fig. 15. Finally, at SNR = 0.85 dB, the invariant curve loses its stability and the turbo decoding algorithm is able to find an unequivocal fixed point.

In the following, we present a general stopping criterion and demonstrate its effectiveness.

#### A. Stopping Criterion

The proposed stopping criterion uses the extrinsic information  $Q_2$  to distinguish among different types of phase trajectories. It divides the hard decisions implied by  $Q_2$  into the following four categories.

- Category 1:** Unreliable hard decisions that are 0.
- Category 2:** Unreliable hard decisions that are 1.
- Category 3:** Reliable hard decisions that are 0.
- Category 4:** Reliable hard decisions that are 1.

Here, we consider a hard decision  $\hat{b}$  on the  $i$ th information bit  $b_i$  to be reliable if  $\Pr_{Q_2}(b_i = \hat{b})$  is greater than a threshold value  $p_\alpha$ . Otherwise, the hard decision is considered unreliable. Let  $N_1, N_2, N_3$ , and  $N_4$ , respectively, be the number of hard decisions in the four categories mentioned above. The proposed criterion stops the decoding algorithm if any of the following three trigger events happens.

- 1)  $N_1, N_2, N_3$ , and  $N_4$  do not change in consecutive iterations.
- 2) Unreliable hard decisions exceed a certain fraction (say,  $U$  percent) of all the hard decisions.
- 3) A predetermined maximum number of iterations (say, MAX iterations), have been performed.

The first trigger event anticipates the convergence of the turbo decoding algorithm and halts the decoding iterations. This trigger event does not imply that the algorithm has converged, rather, it is a good indicator that the algorithm is in the vicinity of a fixed point. Since the extrinsic information for an unequivocal fixed point takes extreme values and the decoding algorithm moves monotonically toward an unequivocal fixed point, iterations after the first trigger event are not likely to change hard decisions based on  $P_0 + Q_1 + Q_2$ . On the other hand, if the algorithm is converging to an indecisive fixed point, more iterations may change a few unreliable hard

decisions. This change is insignificant though, since indecisive fixed points correspond to a large number of erroneous hard decisions.

The second trigger event identifies the phase trajectories that either are converging slowly to an indecisive fixed point or have reached an invariant set. In both cases, the phase trajectory corresponds to a large number of unreliable hard decisions. Note that in the transient period, even a phase trajectory converging to an unequivocal fixed point may have a large number of unreliable hard decisions. Therefore, the second trigger event is used only after a certain minimum number of iterations (say,  $I$  iterations) have been performed. Waiting for  $I$  iterations also gives the phase trajectories going toward an indecisive fixed point a chance to converge.

Finally, there are some exceptional cases, especially in the waterfall region, which neither converge in a reasonable number of iterations nor have a large number of unreliable hard decisions. In such cases, the third trigger event stops the algorithm after MAX iterations.

#### B. Performance of the Proposed Stopping Criteria

The proposed stopping criterion is computationally efficient and simple to implement. At each iteration, it requires approximately  $(\log_2 4)n$  binary operations to categorize the extrinsic information  $Q_2$ . In addition, it requires approximately  $n$  integer operations to compute  $N_1, N_2, N_3$ , and  $N_4$ , and to compare them with different quantities in order to determine trigger events. The proposed criterion stores four integers  $N_1, N_2, N_3$ , and  $N_4$  which are updated during each iteration. These computational resources are at par or better than those required by the CE-criterion and its variants. For example, the modified CE-criterion proposed by Shao, Fossorier, and Lin [14] requires about the same number of arithmetic operations. However, their version of the CE-criterion needs to store approximately  $n$  integers.

Figs. 26 and 27, respectively, show the performance of our stopping criterion and the average number of iterations used by it. We used a turbo code produced by the generator polynomials (7, 5) and an interleaver of length 900. Same parameters were chosen in [8] and [14] to demonstrate the performance of the CE-criterion. We did not fine-tune our stopping criterion for a specific range of SNRs and used the following generic parameters values:

$$\begin{aligned} p_\alpha &= 0.6 \\ I &= 7 \\ \text{MAX} &= 15 \\ U &= 10. \end{aligned}$$

Fig. 26 compares the performance of our stopping criterion with that of the traditional stopping criterion. Here, the number of iterations performed by the traditional stopping criterion for each SNR was obtained by rounding up the *average* number of iterations required by our stopping criterion at that SNR. It is evident from the figure that the proposed stopping criterion results in an additional gain of a few tenths of decibels for bit-error rates less than  $10^{-3}$ .

Fig. 27 shows the average number of iterations required by the proposed stopping criterion. Initially, the average number of

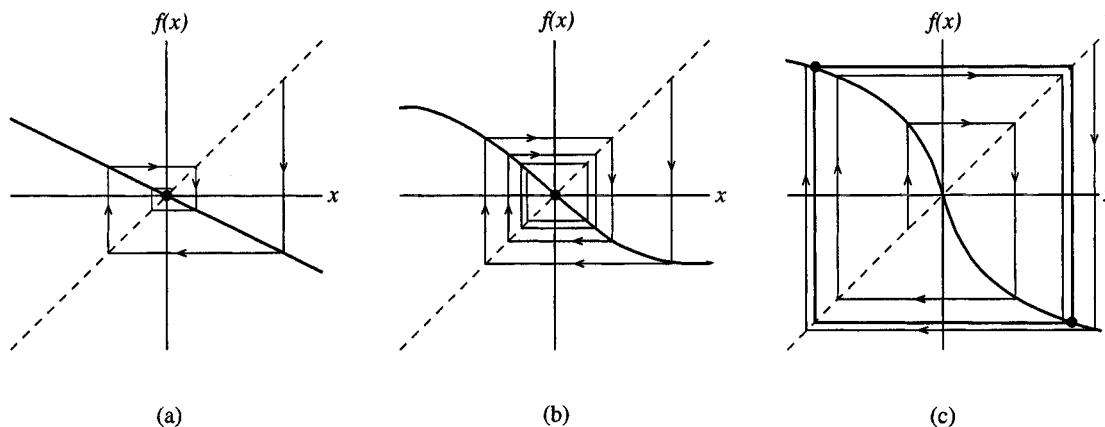


Fig. 16. This figure shows typical changes in a phase trajectory induced by the flip bifurcation of a stable fixed point. (a) Before the bifurcation, phase trajectories jump from one side of the fixed point to another in consecutive iterations until they converge to the fixed point. (b) When an eigenvalue reaches  $-1$ , the fixed point remains stable. However, the phase trajectories take longer to converge. (c) On changing the system parameter further, the eigenvalue moves out of the unit circle. The fixed point now becomes unstable, and is surrounded by a cycle of period two. The phase trajectories that start in the neighborhood of the fixed point now converge to this cycle.

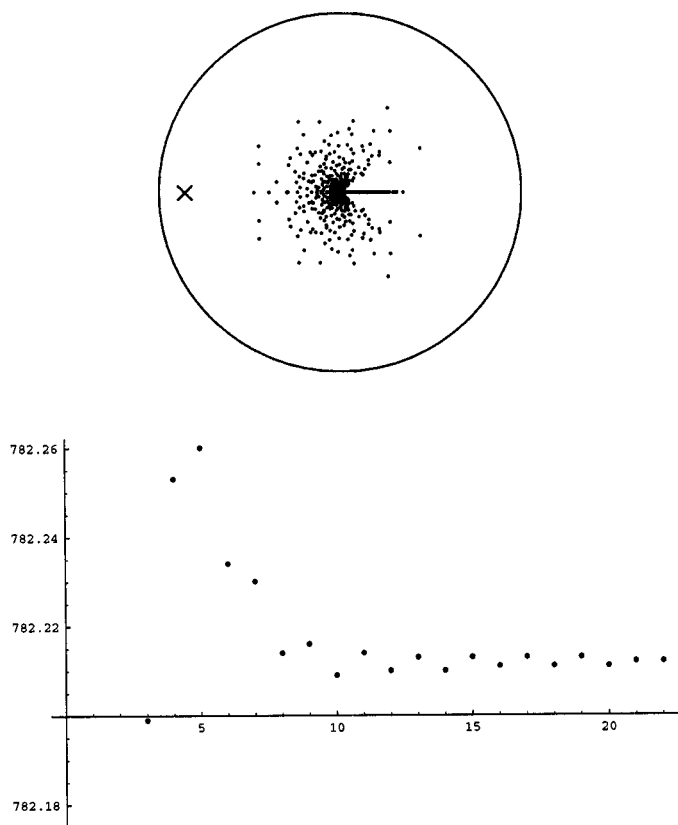


Fig. 17. At SNR = 0.11 dB, the turbo decoding algorithm converges to a stable fixed point. The eigenvalues evaluated at the fixed point lie inside the unit circle, although an eigenvalue is close to  $-1$ .

iterations increase due to the phase trajectories that either converge slowly to an unequivocal fixed point or reach an invariant set with only a few unreliable hard decisions. As the SNR increases, unequivocal fixed points and the associated invariant sets disappear, and the phase trajectories converge faster. There-

fore, the average number of iterations taken by the proposed stopping criterion decreases with the SNR.

This performance compares favorably with the performance of the CE-criterion. The proposed criterion requires an average of 4.92, 3.95, and 3.35 iterations at SNRs 2.0, 2.5, and 3.0 dB, respectively, while the CE-criterion requires 4.87, 3.51, and 2.74 iterations, respectively. Thus, on the average, the CE-criterion requires slightly fewer iterations. However, unlike the CE-criterion [8], which results in up to a 7% increase in the bit-error rate relative to the bit-error rate for 10 iterations, our criterion does not increase the bit-error rate. The average number of iterations required by our criterion can be reduced further by relaxing the conditions for the first trigger event. For example, by stopping iterations when the change in  $N_1$ ,  $N_2$ ,  $N_3$ , and  $N_4$  is less than  $0.003n$ , we can reduced average number of iterations to 4.52, 3.60, and 3.14, respectively, while increasing the bit-error rate by less than 5%. For a specific range of SNRs, our stopping criterion can be fine-tuned further to reduce the average number of iterations. Such fine-tuning is, however, outside the scope of this work.

### VIII. DISCUSSION AND CONCLUSION

Our results in this paper show that the entire SNR range can be subdivided into three main regions with the waterfall region in the middle. For SNRs well below those in the waterfall region, the turbo decoding algorithm has unequivocal as well as indecisive fixed points. Note however, that for asymptotically low SNRs, the decoding algorithm has a unique indecisive fixed point and our analysis strongly implies (though it falls short of proving it) that at asymptotically low SNRs the decoding algorithm should converge to this unique indecisive fixed point. Our simulation results show that even though unequivocal fixed points exist for SNRs well below those in the waterfall region, the turbo decoding algorithm converges to an indecisive fixed point for such SNRs. As the SNR increases and approaches the waterfall region, indecisive fixed points bifurcate. The bifurca-

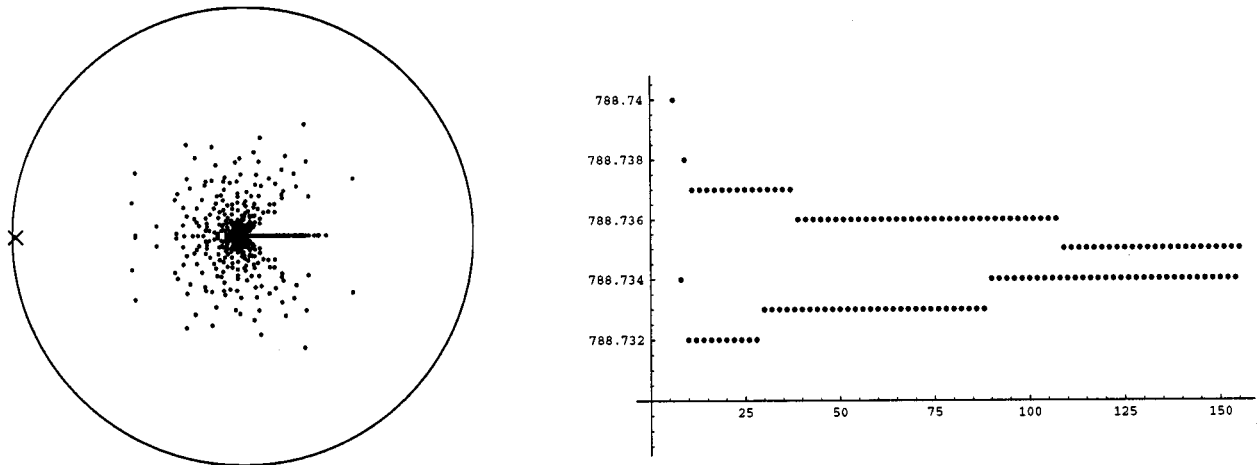


Fig. 18. As the SNR is increased to 0.132 dB, the eigenvalue moves closer to  $-1$  and the turbo decoding algorithm takes longer to converge.

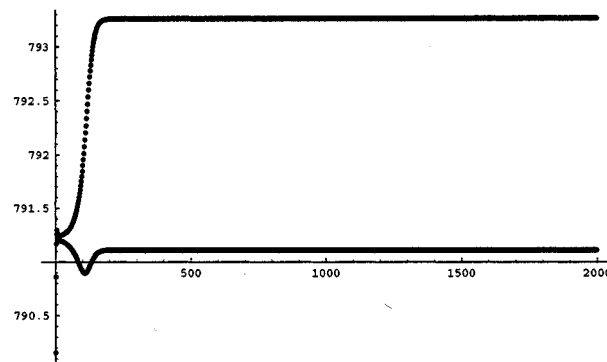


Fig. 19. At SNR = 0.14 dB, the fixed point has undergone a flip bifurcation and is unstable. As a result, the phase trajectory ends up in a stable cycle of period two.

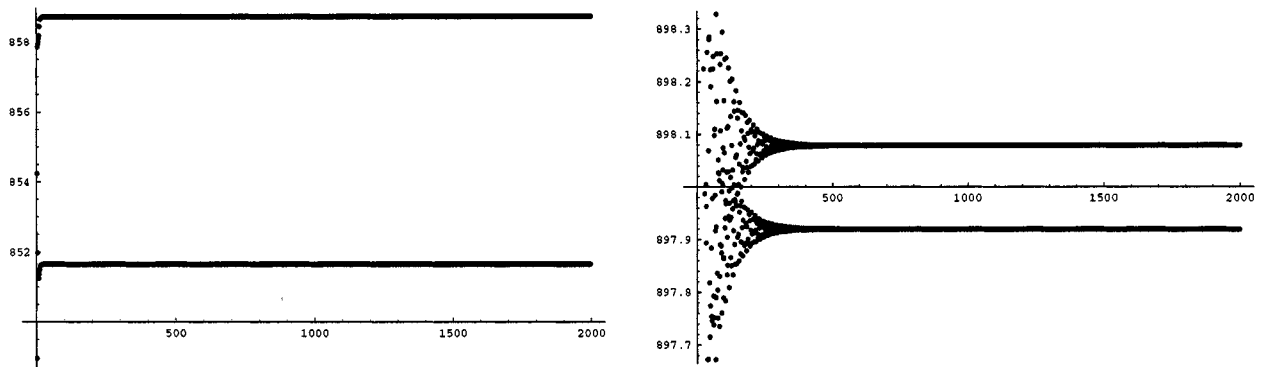


Fig. 20. This cycle remains stable at SNR = 0.40 and 0.60 dB. Note that as the SNR increases, the cycle moves toward the correct solution from 855 to 897.5.

tion of indecisive fixed points often produces asymptotically stable invariant sets. The phase trajectories that start from the unbiased initialization now get stuck in these invariant sets, and

as a consequence, the turbo decoding algorithm may never converge for SNRs in the waterfall region. As the SNR increases even further, the invariant sets either become unstable or disap-

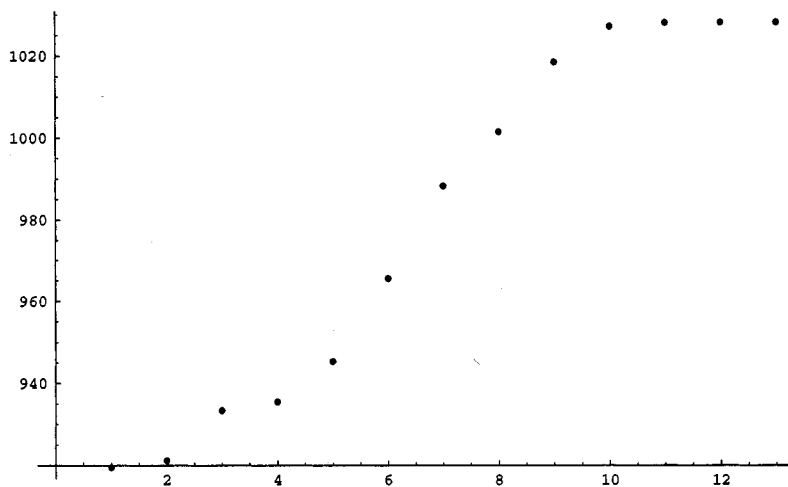


Fig. 21. Finally, at SNR = 0.70 dB, the cycle of period two loses its stability and the turbo decoding algorithm is able to find an unequivocal fixed point.

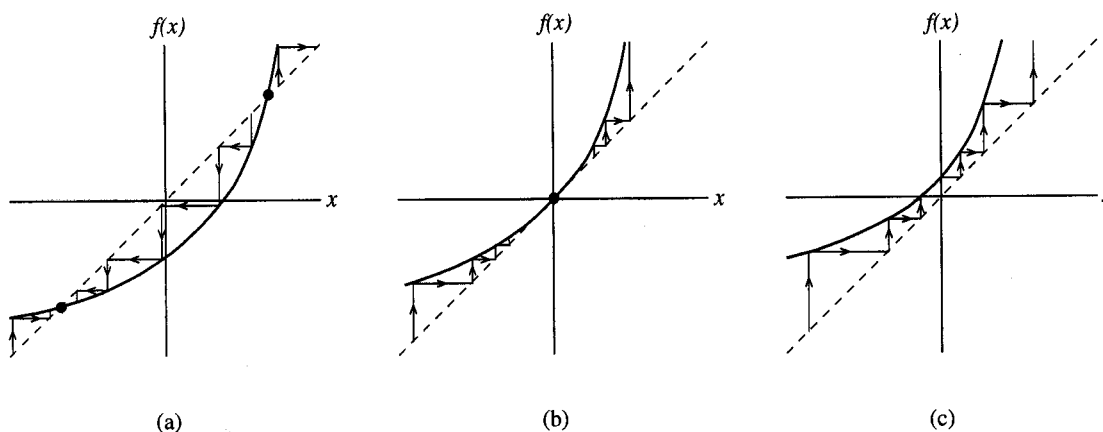


Fig. 22. This figure shows typical changes in a phase trajectory induced by the fold bifurcation of a fixed point. (a) Before the bifurcation, phase trajectories converge to the fixed point. (b) On changing the system parameter, when an eigenvalue reaches +1, the fixed point becomes unstable. Since the fold bifurcation does not result in an invariant set, phase trajectories starting close to the fixed point wander away from its neighborhood. (c) On changing the parameter further, the fixed point disappears.

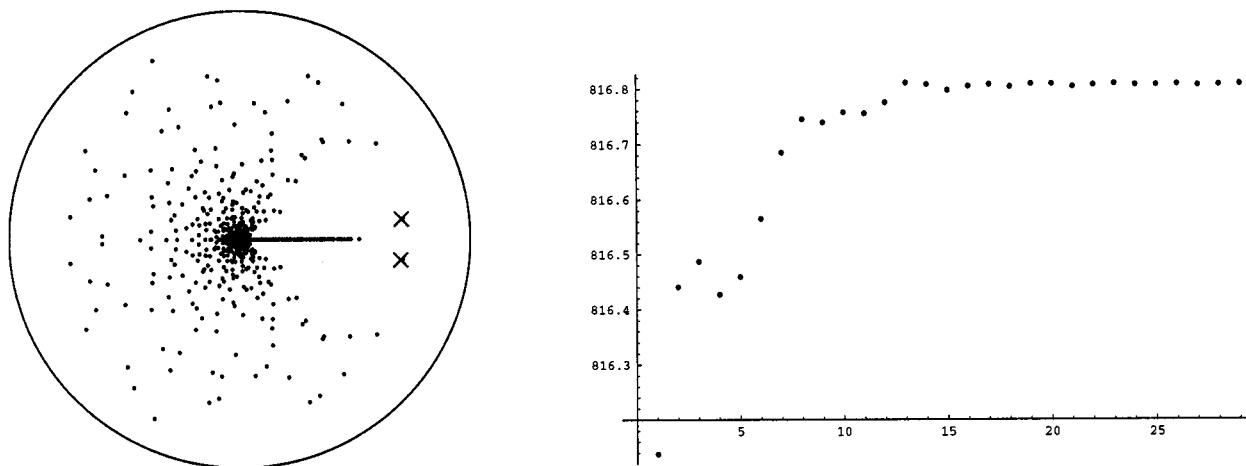


Fig. 23. At SNR = 0.72 dB, a pair of eigenvalues is close to the unit circle. The turbo decoding algorithm converges to a stable fixed point.

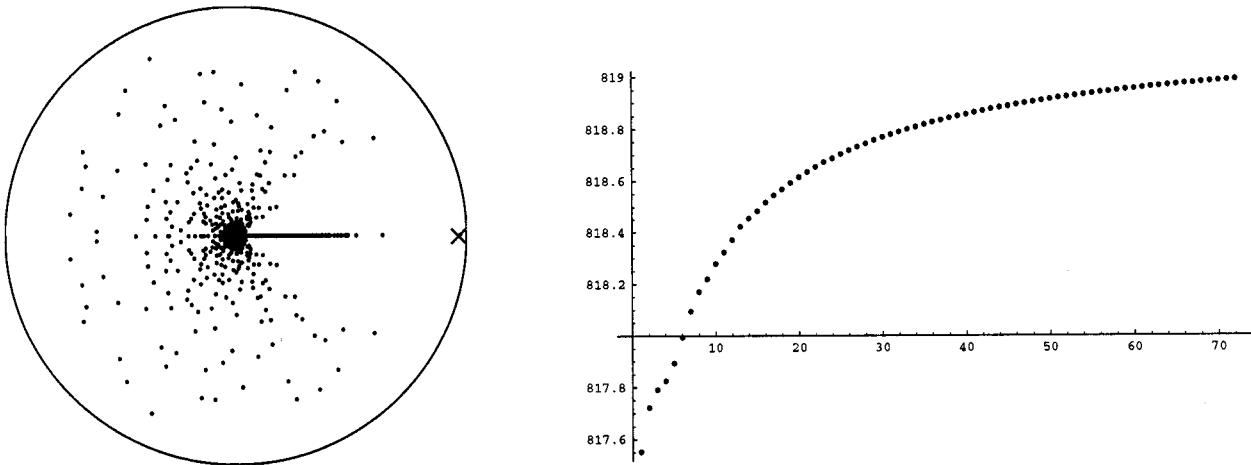


Fig. 24. As the SNR is increased to 0.7244 dB, the pair of complex conjugate eigenvalues splits and one of the eigenvalue moves very close to the unit circle. As a result, the turbo decoding algorithm takes much longer to converge.

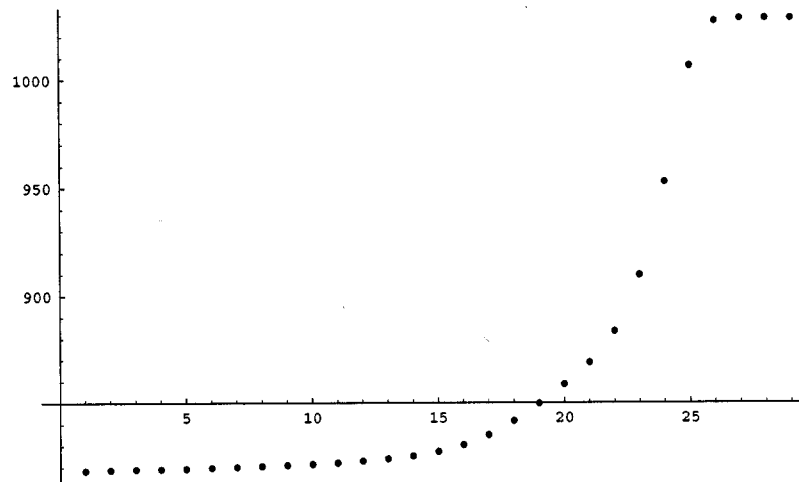


Fig. 25. Increasing the SNR to 0.73 dB causes the fixed point to undergo a fold bifurcation. Since the fold bifurcation does not result in an invariant set, the turbo decoding algorithm is able to find an unequivocal fixed point.

pear. For SNRs higher than those in the waterfall region, neither indecisive fixed points nor invariant sets exist, and the turbo decoding algorithm converges to an unequivocal fixed point.

A phase trajectory converging to an unequivocal fixed point has characteristics that differ significantly from those of other phase trajectories. When a phase trajectory fails to converge to an unequivocal fixed point, these differences can easily be identified and used to declare a *decoding failure*. This extends the use of turbo codes in transmission schemes that employ automatic repeat request (ARQ) protocols to retransmit a codeword. The ability to declare a decoding failure also eliminates a frequently cited shortcoming of the turbo codes with respect to the Gallager's low-density parity-check codes. It is well known that a decoding algorithm for low-density parity-check codes can declare a failure if there are decoding errors in the output [10]. Using the characteristics of its phase trajectory, the turbo de-

coding algorithm can also do the same without any significant increase in the implementation complexity.

The unequivocal fixed points of the turbo decoding algorithm are quite stable. Typically, the Jacobian, evaluated at an unequivocal fixed point, has eigenvalues of magnitude less than  $1/3$ . This indicates that unequivocal fixed points have large domains of attraction, and phase trajectories that start in the neighborhood of an unequivocal fixed point are strongly attracted toward it. As a consequence, even if some noise is added at each iteration, such phase trajectories will remain in the neighborhood of the fixed point, provided that the magnitude of noise is not too large.

In hindsight, strong stability of unequivocal fixed points provides a justification for using suboptimal algorithms in place of the optimal BCJR algorithm. A suboptimal algorithm can be thought of as introducing some noise in a phase trajectory

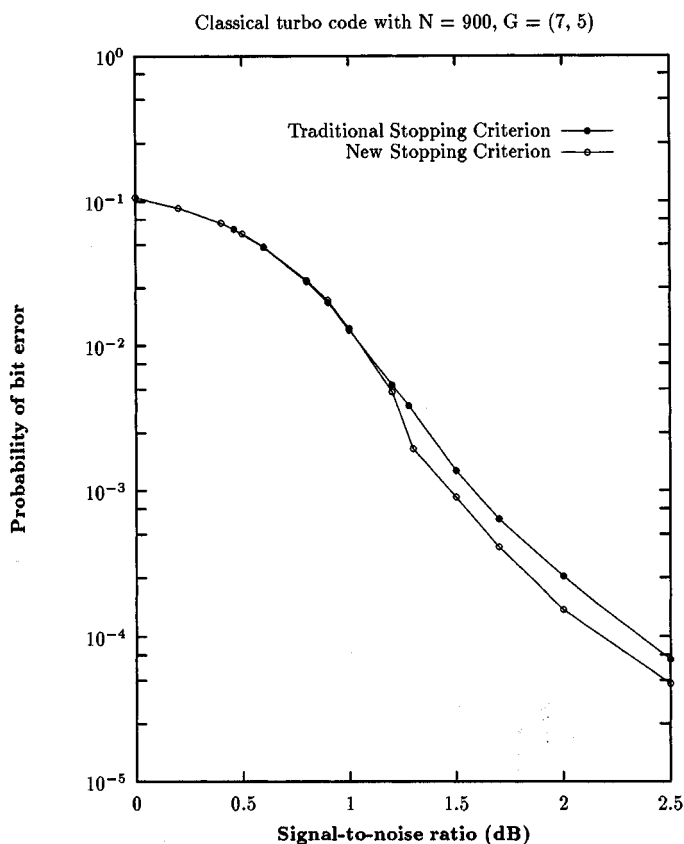


Fig. 26. The performance of the turbo decoding algorithm with the proposed and traditional stopping criteria.

at each iteration. When a phase trajectory converges toward an unequivocal fixed point, this noise is unlikely to affect the final hard decisions. On the other hand, when a phase trajectory converges toward an indecisive fixed point, a suboptimal algorithm may cause the divergence of the phase trajectory. In that case, the performance of the suboptimal algorithm will be close to the performance of the BCJR algorithm, since an indecisive fixed point often corresponds to a bit-error rate that is only slightly less than the uncoded bit-error rate. Indeed, Hagenauer, Offer, and Papke have shown that using a suboptimal algorithm results only in a small loss of performance [8].

Serially concatenated systems often employ an inner convolutional code in conjunction with an outer block code. It is tempting to replace the inner convolutional code in a concatenated system by a turbo code. However, the results herein show that such a replacement is likely to be useless. In contrast to convolutional codes, where the number of decoding errors in a transmitted block takes a wide range of values, the number of errors for a turbo codeword is either very small or very large. In case there are only a few decoding errors, the outer code will yield a small improvement. On the other hand, if there are a large number of decoding errors, the outer code will simply fail. An outer code is however useful at high SNRs, when the number of errors in a decoded codeword is typically very small. In such cases, a high-rate outer block code can be used to lower the error floor of turbo codes.

In this paper, we considered the AWGN channel. Analytical results presented here (especially in Section IV) depend only

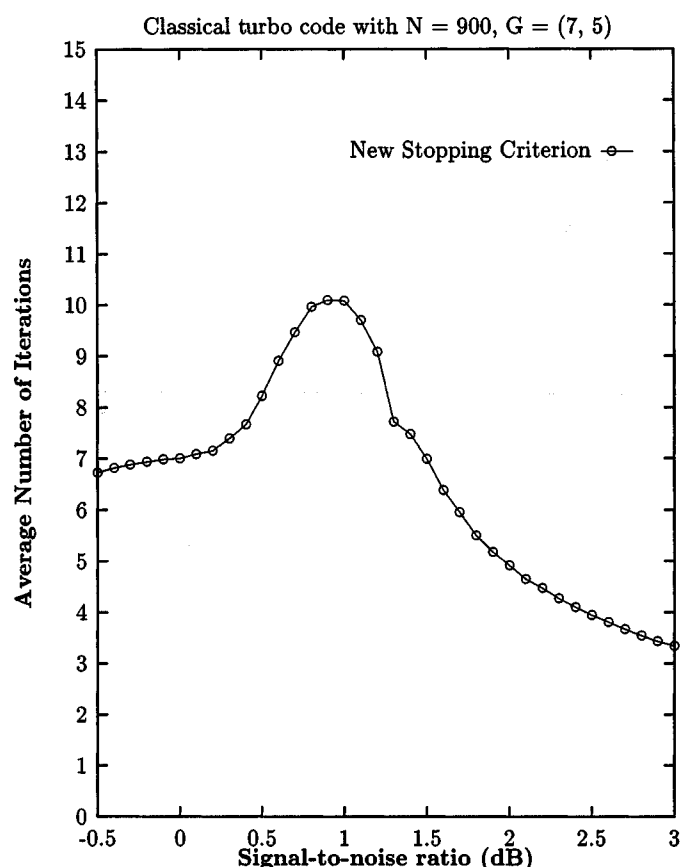


Fig. 27. The average number of iterations done by the proposed stopping criterion.

on the memoryless property of the channel and on some bounds on  $P_0$ ,  $P_1$ , and  $P_2$ . These bounds are natural and are satisfied by most noise distributions at asymptotically low and high SNRs. Therefore, at asymptotic SNRs, the properties of fixed points for such noise distributions will be the same as shown here for Gaussian noise. We conjecture that even at practical SNRs, phase trajectories for other channels will have the characteristics similar to those presented here for the AWGN channel.

Some recent results by Richardson and Urbanke suggest that for many iterative decoding algorithms, there is a single threshold that divides SNRs into two regions, each with distinct phase trajectories [13]. These results are derived for codes with asymptotically large lengths. Experimental results in this paper suggest that for codes of finite length, there is a region of transition over which characteristics of phase trajectories change from one type to another. As the length of the code becomes larger, this transition region becomes smaller, and in the limit it disappears. Thus, the results in this paper are complementary, rather than contradictory, to the results obtained by Richardson and Urbanke.

APPENDIX A  
PROOF OF LEMMA 3

*Lemma 4:* Given an  $\epsilon$  between 0 and 1, there exists a  $\sigma(\epsilon)$  such that for  $0 < \sigma < \sigma(\epsilon)$ , the probability that  $(\tilde{x}, \tilde{y}, \tilde{z}) \in \mathfrak{B}$  is at least  $1 - \epsilon$ .

*Proof:* Let  $\mathfrak{Z}_1$  be the set of all received vectors  $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\mathbf{z}})$  that satisfy the first condition for  $\mathfrak{Z}$  (see Section IV-B). Similarly, let  $\mathfrak{Z}_2$  be the set of all received vectors  $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\mathbf{z}})$  that satisfy the second condition. Clearly,  $\mathfrak{Z} = \mathfrak{Z}_1 \cap \mathfrak{Z}_2$ , and

$$\Pr(\mathfrak{Z}) \geq 1 - \Pr(\mathfrak{Z}_1^c) - \Pr(\mathfrak{Z}_2^c)$$

Hence, it suffices to show that  $\Pr(\mathfrak{Z}_1^c)$  and  $\Pr(\mathfrak{Z}_2^c)$  can be made arbitrarily small by choosing  $\sigma > 0$  small enough. We will show this only for  $\Pr(\mathfrak{Z}_1^c)$ , since the proof for the other counterpart is similar.

First, consider the probability of  $p_0(\mathbf{b}^i) \geq \frac{1}{2^n}$ . In the beginning of Section IV, we showed that for AWGN channel with BPSK modulation,  $P_0(\mathbf{b}^i)$  is a Gaussian random variable with mean  $-\frac{2}{\sigma^2}$  and variance  $\frac{4}{\sigma^2}$ . Therefore, it follows that

$$\begin{aligned} \Pr\left(p_0(\mathbf{b}^i) \geq \frac{1}{2^n}\right) &= \Pr(P_0(\mathbf{b}^i) \geq -n \log 2) \\ &= Q\left(\frac{-n \log 2 + \frac{2}{\sigma^2}}{\frac{2}{\sigma}}\right) \\ &= Q\left(\frac{2 - n\sigma^2 \log 2}{2\sigma}\right). \end{aligned} \quad (39)$$

Since  $Q(x)$  decreases monotonically to zero, we can choose a  $\sigma_{0i}(\delta)$  such that, for  $0 < \sigma < \sigma_{0i}(\delta)$

$$\Pr\left(p_0(\mathbf{b}^i) \geq \frac{1}{2^n}\right) \leq \delta. \quad (40)$$

Next, consider the probability of  $p_j(\mathbf{b}^i) \geq \frac{1}{4}$  for  $j = 1$  and  $2$ , and  $i = 1, \dots, n$ . We assume that the constituent convolutional codes are chosen in such a manner that for all values of  $i$  and  $j$ ,  $\text{wt}(\mathbf{c}_j(\mathbf{b}^i)) > 0$ . This is certainly true for the classical turbo codes. Under this assumption, using a similar argument as above, we can show that there exists a  $\sigma_{ji}(\delta)$  such that for  $0 < \sigma < \sigma_{ji}(\delta)$

$$\Pr\left(p_j(\mathbf{b}^i) \geq \frac{1}{2}\right) \leq \delta. \quad (41)$$

Let

$$\sigma_1(\epsilon) = \min_{\substack{1 \leq i \leq n \\ 0 \leq j \leq 2}} \sigma_{ij}\left(\frac{\epsilon}{6n}\right).$$

Then, using the union bound on  $\mathfrak{Z}_1$  along with (40) and (41)

$$\begin{aligned} \Pr(\mathfrak{Z}_1^c) &\leq \sum_{i=1}^n \Pr\left(\left\{p_0(\mathbf{b}^i) \geq \frac{1}{2^n}\right\}\right) \\ &\quad + \sum_{i=1}^n \Pr\left(\left\{p_1(\mathbf{b}^i) \geq \frac{1}{2}\right\}\right) \\ &\quad + \sum_{i=1}^n \Pr\left(\left\{p_2(\mathbf{b}^i) \geq \frac{1}{2}\right\}\right) \\ &\leq n \frac{\epsilon}{6n} + n \frac{\epsilon}{6n} + n \frac{\epsilon}{6n}, \quad \text{for } 0 < \sigma < \sigma_1(\epsilon) \\ &= \frac{\epsilon}{2}. \end{aligned} \quad (42)$$

A similar argument shows that there exists a  $\sigma_2(\epsilon)$ , such that for  $0 < \sigma < \sigma_2(\epsilon)$ ,  $\Pr(\mathfrak{Z}_2^c) \leq \epsilon/2$ . We finish the proof by setting  $\sigma(\epsilon) = \min\{\sigma_1(\epsilon), \sigma_2(\epsilon)\}$ .  $\square$

## APPENDIX B

### AN ALTERNATIVE FORMULATION OF THE TURBO DECODING ALGORITHM

In this appendix, we present an alternative formulation of the turbo decoding algorithm that eliminates the redundancy present in (15) and (16). We start with a rewrite of (15) and (16) to include the iteration number in the system equations

$$Q_1^{(l)} = \pi_{P_1}\left(P_0 + Q_2^{(l-1)}\right) - \left(P_0 + Q_2^{(l-1)}\right) \quad (43)$$

$$Q_2^{(l)} = \pi_{P_2}\left(P_0 + Q_1^{(l)}\right) - \left(P_0 + Q_1^{(l)}\right). \quad (44)$$

Let  $P_t^{(l)} = P_0 + Q_1^{(l)} + Q_2^{(l)}$  be the total posterior density on  $\mathcal{H}$ . Since the projection map  $\pi_P$  is bijective [12], using (44), we get

$$\begin{aligned} Q_1^{(l)} &= \pi_{P_2}^{-1}\left(P_t^{(l)}\right) - P_0 \\ &:= \Theta\left(P_t^{(l)}\right). \end{aligned} \quad (45)$$

It is easy to see that the function  $\Theta$  is bijective and has a well-defined inverse. Similarly,  $Q_2^{(l)}$  can be specified in terms of  $P_t^{(l)}$  as follows:

$$\begin{aligned} Q_2^{(l)} &= P_t^{(l)} - \left(P_0 + Q_1^{(l)}\right) \\ &= P_t^{(l)} - \pi_{P_2}^{-1}\left(P_t^{(l)}\right) \\ &:= \Psi\left(P_t^{(l)}\right). \end{aligned} \quad (46)$$

Substituting (45) and (46) in (43), we obtain

$$\begin{aligned} \Theta\left(P_t^{(l)}\right) &= \pi_{P_1}\left(P_0 + \Psi\left(P_t^{(l-1)}\right)\right) - \left(P_0 + \Psi\left(P_t^{(l-1)}\right)\right) \\ P_t^{(l)} &= \Theta^{-1}\left[\pi_{P_1}\left(P_0 + \Psi\left(P_t^{(l-1)}\right)\right) - \left(P_0 + \Psi\left(P_t^{(l-1)}\right)\right)\right] \\ &= \pi_{P_2}\left(\pi_{P_1}\left[P_0 + \Psi\left(P_t^{(l-1)}\right)\right] - \Psi\left(P_t^{(l-1)}\right)\right) \end{aligned} \quad (47)$$

$$:= \mathcal{F}\left(P_t^{(l-1)}\right). \quad (48)$$

The function  $\mathcal{F}$  defines an alternative dynamical system which iterates on the  $n$ -dimensional variable  $P_t$ . This dynamical system is fundamentally the same as the system described by (15) and (16). In particular, if  $P_t^*$  is a fixed point of  $\mathcal{F}$ , then  $Q_1^* = \Theta(P_t^*)$  and  $Q_2^* = \Psi(P_t^*)$  is a fixed point of (15) and (16).

The stability of a fixed point of  $\mathcal{F}$  is determined by the eigenvalues of its Jacobian evaluated at the fixed point. Recall that the Jacobian of a function  $f: \mathbb{R}^n \rightarrow \mathbb{R}^n$  is given by an  $n \times n$  matrix  $Df$

$$(Df)_{i,j} = \frac{\partial f_i}{\partial x_j} \quad (49)$$

where  $f_i$  and  $x_i$  are the  $i$ th coordinates of  $f$  and  $x$ , respectively. In the rest of this appendix, the Jacobian  $D\mathcal{F}$  will be computed. This Jacobian was used to demonstrate the bifurcation of indecisive fixed points in Section VI.

Let

$$P_t^* = P_0 + Q_1^* + Q_2^*$$

be a fixed point of  $\mathcal{F}$ , where  $Q_1^* = \Theta(P_t^*)$  and  $Q_2^* = \Psi(P_t^*)$ . Let  $J_P(X)$  denote the Jacobian of  $\pi_P$  evaluated at the product log-density  $X$ . Richardson [12] showed that

$$(J_P(\mathbf{0}))_{ij} = \Pr_P(\mathcal{H}_j|\mathcal{H}_i) - \Pr_P(\mathcal{H}_j|\mathcal{H}_i^c) \quad (50)$$

where  $\Pr_P$  is the probability measure on  $\mathcal{H}$  induced by the log-density  $P$ .

By the chain-rule of derivatives, we can write the Jacobian  $D\mathcal{F}$  in terms of  $J_P$

$$\begin{aligned} D\mathcal{F}(P_t^*) &= J_{P_2}(\pi_{P_1}(P_0 + \Psi(P_t^*)) - \Psi(P_t^*)) \\ &\quad \cdot [J_{P_1}(P_0 + \Psi(P_t^*))D\Psi(P_t^*) - D\Psi(P_t^*)] \\ &= J_{P_2}(\pi_{P_1}(P_0 + Q_2^*) - (P_0 + Q_2^*) + P_0) \\ &\quad \cdot [J_{P_1}(P_0 + Q_2^*) - I]D\Psi(P_t^*). \end{aligned} \quad (51)$$

Since  $Q_1^*$  and  $Q_2^*$  satisfy (43), we have

$$D\mathcal{F}(P_t^*) = J_{P_2}(Q_1^* + P_0)[J_{P_1}(P_0 + Q_2^*) - I]D\Psi(P_t^*). \quad (52)$$

Next, we take the derivative of the identity  $\Psi(X) = X - \pi_{P_2}^{-1}(X)$  and obtain  $D\Psi(P_t^*)$

$$\begin{aligned} D\Psi(P_t^*) &= I - J_{P_2}^{-1}(\pi_{P_2}^{-1}(P_t^*)) \\ &= I - J_{P_2}^{-1}(P_0 + \Theta(P_t^*)) \end{aligned} \quad (53)$$

$$= I - J_{P_2}^{-1}(P_0 + Q_1^*). \quad (54)$$

Substituting the value of  $D\Psi(P_t^*)$  in (52), we get

$$\begin{aligned} D\mathcal{F}(P_t^*) &= J_{P_2}(Q_1^* + P_0)(J_{P_1}(P_0 + Q_2^*) - I)(I - J_{P_2}^{-1}(P_0 + Q_1^*)) \end{aligned} \quad (55)$$

Finally, since the eigenvalues of  $ABC$  are equal to the eigenvalues of  $BCA$ , the stability of a fixed point  $P_t^*$  can be determined by the eigenvalues of

$$(J_{P_1}(P_0 + Q_2^*) - I)(J_{P_2}(P_0 + Q_1^*) - I).$$

We further remark that  $J_P(Q) = J_{P+Q}(\mathbf{0})$ , and  $J_{P+Q}(\mathbf{0})$  can be computed by modifying the BCJR algorithm.

## APPENDIX C

### CENTER MANIFOLD THEOREM AND THE REDUCTION PRINCIPLE

Consider a discrete dynamical system given by

$$x \mapsto f(x), \quad x \in \mathbb{R}^n \quad (56)$$

where  $f$  is a smooth (infinitely differentiable) function with  $f(\mathbf{0}) = \mathbf{0}$ . Let  $\mu_1, \dots, \mu_n$  be the eigenvalues of the Jacobian  $Df$  evaluated at the fixed point  $\mathbf{0}$ . Assume that among  $\mu_1, \dots, \mu_n, n_c$  eigenvalues (counting multiplicities) have magnitude 1. Let  $E^c$  be the *generalized* linear eigenspace of  $Df$  that corresponds to the eigenvalues with magnitude 1. Using an eigenbasis, we can (locally) linearize the system (56) as follows:

$$\begin{pmatrix} u \\ v \end{pmatrix} \mapsto \begin{pmatrix} Bu + g(u, v) \\ Cv + h(u, v) \end{pmatrix}, \quad u \in \mathbb{R}^{n_c}, v \in \mathbb{R}^{n-n_c}. \quad (57)$$

Here, the Taylor series expansions of  $g$  and  $h$  start from the quadratic terms, that is,  $g(\mathbf{0}, \mathbf{0}) = h(\mathbf{0}, \mathbf{0}) = \mathbf{0}$ , and  $Dg(\mathbf{0}, \mathbf{0}) = Dh(\mathbf{0}, \mathbf{0}) = \mathbf{0}$ . Note that all eigenvalues of  $B$  have magnitude 1, while none of the eigenvalues of  $C$  has magnitude 1.

An  $n_c$ -dimensional *invariant* manifold, defined in a small neighborhood of  $\mathbf{0}$ , is referred to as a *center manifold* provided

that it passes through  $\mathbf{0}$  and is tangent to  $E^c$  at  $\mathbf{0}$ . It follows that a center manifold  $W^c$  can be locally represented as a graph of a smooth function  $V$  [5]

$$W^c = \{(u, v) : v = V(u)\}. \quad (58)$$

Here  $V: \mathbb{R}^{n_c} \rightarrow \mathbb{R}^{n-n_c}$ . The Center Manifold Theorem [9], [16] is an existence theorem which guarantees the existence of a center manifold for (56).

*Theorem 4 (Center Manifold Theorem):* There exists a center manifold for (56) with the same finite smoothness as  $f$ . Moreover, in sufficiently small neighborhoods of  $\mathbf{0}$ , the dynamics of (56) restricted to the center manifold is given by the following  $n_c$ -dimensional system:

$$u \mapsto Bu + g(u, V(u)), \quad u \in \mathbb{R}^{n_c} \quad (59)$$

The next theorem, referred to as the Reduction Principle, implies that the dynamics of (59) near the point  $u = \mathbf{0}$  determines the dynamics of (56) near the point  $x = \mathbf{0}$  [9], [16].

*Theorem 5 (The Reduction Principle):* Near the origin, the system (56) is locally topologically equivalent to the system

$$\begin{pmatrix} u \\ v \end{pmatrix} \mapsto \begin{pmatrix} Bu + g(u, V(u)) \\ Cv \end{pmatrix}. \quad (60)$$

The topological equivalence of two dynamical systems implies that the phase trajectories of one system are related to the phase trajectories of another by a homeomorphic transformation. Therefore, their phase trajectories are “qualitatively similar.”

Notice that the variables  $u$  and  $v$  are decoupled in (60). The first equation is a restriction of (57) to its center manifold, while the second equation represents a linear system. The reduction principle provides considerable simplifications in studying the asymptotic behavior of phase trajectories in a small neighborhood of a fixed point. For example, if none of the eigenvalues of  $Df$  at  $\mathbf{0}$  has modulus greater than 1, then all eigenvalues of  $C$  will have modulus less than 1. It follows that in a neighborhood of  $\mathbf{0}$ , asymptotically  $v \rightarrow \mathbf{0}$ , and by the reduction principle (56) evolves just like an  $n_c$ -dimensional system given by (59).

Next, consider a stable fixed point of an  $n$ -dimensional smooth system dependent on a single parameter  $\alpha$ . Since the system is smooth, the fixed point as well as the Jacobian evaluated at this point is a continuous function of the system parameter. Furthermore, since the eigenvalues are continuous functions of the Jacobian, it follows that they also vary continuously with the system parameter.

Initially, since the fixed point under consideration is stable, all eigenvalues of the Jacobian have magnitudes less than 1. However, as the system parameter  $\alpha$  is changed, magnitudes of the eigenvalues may also change, and it is possible that the magnitude of an eigenvalue becomes greater than 1. This will render the fixed point unstable. The change in the stability of a fixed point results in a topologically nonequivalent change in the phase portrait (collection of phase trajectories) of the system. This phenomenon is referred to as the *bifurcation* of a stable fixed point, and the value of the system parameter at which a fixed point bifurcates is referred to as the *critical value*.

After bifurcation, generically, none of the eigenvalues will be on the unit circle, and the center manifold theorem and the reduction principle cannot be applied directly. However, with a slight twist, we can use the reduction principle to simplify the analysis of a parameter-dependent system. Consider the following discrete dynamical system parameterized by a single parameter  $\alpha$ :

$$x \mapsto f(x, \alpha), \quad x \in \mathbb{R}^n, \alpha \in \mathbb{R}. \quad (61)$$

Assume that at the critical value  $\alpha = 0$ , the system has a fixed point at  $x = 0$  with  $n_c$  eigenvalues on the unit circle. Consider the following *extended* system which includes the parameter  $\alpha$  as a new *dependent* variable:

$$\begin{aligned} \alpha &\mapsto \alpha \\ x &\mapsto f(x, \alpha). \end{aligned} \quad (62)$$

Clearly, the extended system (62) has a fixed point at  $(x, \alpha) = (0, 0)$  with  $n_c + 1$  eigenvalues on the unit circle. Thus, we can apply the Center Manifold Theorem to (62) and conclude that there is an  $(n_c + 1)$ -dimensional center manifold  $W^c$  which is tangent at the origin  $(0, 0)$  to the  $n_c + 1$  eigenvectors corresponding to the eigenvalues of unit magnitude. Since the hyperplanes  $\Pi_{\alpha_0} = \{(\alpha, x): \alpha = \alpha_0\}$  are invariant with respect to the system (62), the center manifold  $W^c$  is foliated (partitioned) by the following  $n_c$ -dimensional manifolds:

$$W_\alpha^c = W^c \cap \Pi_\alpha. \quad (63)$$

For the small values of  $\alpha$ , let

$$W_\alpha^c = \{(u, v, \alpha): v = V(u, \alpha)\}$$

be the representation of  $W^c$  as a graph of  $(u, \alpha)$ , where  $u$  belongs to the generalized eigenspace corresponding to the  $n_c$  eigenvalues of  $Df(x, 0)$  with unit magnitude. The reduction principle implies that for small values of  $\alpha$ , the phase trajectories of (62) near  $x = 0$  can be studied by restricting it to  $W_\alpha^c$ . This results in a system of the form

$$u \mapsto Bu + g(u, \alpha, V(u, \alpha)) \quad (64)$$

where all  $n_c$  eigenvalues of  $B$  are on the unit circle and the Taylor series expansion of  $g$  starts from the quadratic terms.

In a generic system, on changing the system parameter, either a single eigenvalue will cross the unit circle through  $\pm 1$  or two complex conjugate eigenvalues will cross the unit circle together. Hence, in a generic system,  $n_c$  equals one or two. Consequently, for the values close to the critical parameter, the phase trajectories near a stable fixed point (for an  $n$ -dimensional generic system) resemble the phase trajectories for a one-

or two-dimensional system [9], [16]. In Section VI, we used this fact to locate bifurcations of stable indecisive fixed points by comparing the asymptotic behaviors of the phase trajectories (associated with indecisive fixed points) with their counterparts in one- and two-dimensional systems. Later, these observations were confirmed by explicitly computing the eigenvalues of the Jacobian evaluated at indecisive fixed points, and by showing that indeed these eigenvalues cross the unit circle, thereby making the indecisive fixed points unstable.

#### ACKNOWLEDGMENT

The authors wish to thank R. Kötter, T. Richardson, and R. Urbanke for stimulating this work in many ways. They thank the referees for useful comments that improved the presentation of paper.

#### REFERENCES

- [1] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284–287, Mar. 1974.
- [2] S. Benedetto and G. Montorsi, "Unveiling turbo codes: Some results on parallel concatenated coding schemes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 409–427, Mar. 1996.
- [3] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Trans. Commun.*, vol. 44, pp. 1261–1271, Oct. 1996.
- [4] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. 1993 IEEE Int. Conf. Communications*, Geneva, Switzerland, 1993, pp. 1064–1070.
- [5] W. M. Boothby, *An Introduction to Differential Manifolds and Riemannian Geometry*, 2nd ed. San Diego, CA: Academic, 1986, vol. 120, Pure and Applied Mathematics.
- [6] D. Divsalar and F. Pollara, "On the design of turbo codes," Jet Propulsion Lab., Pasadena, CA, TDA Progr. Rep. 42-123, 1995.
- [7] L. Duan and B. Rimoldi, "The turbo decoding algorithm has fixed points," in *Proc. IEEE Int. Symp. Information Theory*, Cambridge, MA, Aug. 1998, p. 277.
- [8] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding and binary block and convolutional codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 429–445, Mar. 1996.
- [9] Y. A. Kuznetsov, *Elements of Applied Bifurcation Theory*. New York: Springer-Verlag, 1998, vol. 112, Applied Mathematical Sciences.
- [10] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–431, Mar. 1999.
- [11] L. C. Perez, J. Seghers, and D. J. Costello, "A distance spectrum interpretation of turbo codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 1698–1709, Nov. 1996.
- [12] T. Richardson, "The geometry of turbo-decoding dynamics," *IEEE Trans. Inform. Theory*, to be published.
- [13] T. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599–618, Feb. 2001.
- [14] R. Y. Shao, M. Fossorier, and S. Lin, "Two simple stopping criteria for iterative decoding," in *Proc. IEEE Int. Symp. Information Theory*, Cambridge, MA, Aug. 1998, p. 279.
- [15] N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation, Dept. Elec. Eng., Linköping University, Linköping, Sweden, 1996.
- [16] S. Wiggins, *Introduction to Applied Nonlinear Dynamical Systems and Chaos*. New York: Springer-Verlag, 1990, vol. 2, Texts in Applied Mathematics.