ITERATIVE ROW-COLUMN ALGORITHMS FOR TWO-DIMENSIONAL INTERSYMBOL INTERFERENCE CHANNEL EQUALIZATION: COMPLEXITY REDUCTION AND PERFORMANCE ENHANCEMENT

By

HANNAN MA

A thesis submitted in partial fulfillment of
the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

WASHINGTON STATE UNIVERSITY
School of Electrical Engineering and Computer Science

AUGUST 2010

To the Faculty of Washington State University:

The members of the Committee appointed to examine the thesis of HANNAN MA find it satisfactory and recommend that it be accepted.

Benjamin Belzer, Ph.D., Chair

Krishnamoorthy Sivakumar, Ph.D., Co-Chair

Thomas Fischer, Ph.D.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my advisors, Dr. Benjamin Belzer and Dr. Krishnamoorthy Sivakumar, for their invaluable guidance and consistent support through out the time it took me to complete this research and write the thesis in Washington State University. The program was one of the most important and formative experiences in my life. Without their help this thesis would not have been possible.

A member of my thesis committee, Dr. Thomas Fischer, has generously given his time and expertise to better my work. Courses I have taken are also inspirational and valuable for my current and further research. I thank them for their contribution and their good-natured support.

The work in this dissertation was partially supported by the National Science Foundation under grant CCF-0635390. I want to express my appreciation to NSF for their financial support.

I also wish to thank all the staff of the School of Electrical Engineering and Computer Science, for their help and support.

Finally, I would like to express my gratitude to my parents, Sakuya and Senjogahara Hitagi, for their love, support, patience and encouragement.

ITERATIVE ROW-COLUMN ALGORITHMS FOR TWO-DIMENSIONAL INTERSYMBOL INTERFERENCE CHANNEL EQUALIZATION: COMPLEXITY REDUCTION AND PERFORMANCE ENHANCEMENT

Abstract

by Hannan MA, M.S.
Washington State University
August 2010

Chair: Benjamin Belzer, Krishnamoorthy Sivakumar (Co-Chair)

This thesis uses the iterative row-column soft-decision feedback algorithm (IRCSDFA) of Cheng et.al. IEEE Sig. Proc. Letters 2007 as a starting point; comparisons in that work show that the IRCSDFA is one of the leading algorithms for turbo equalization of two dimensional inter symbol interference (ISI) channels with additive white Gaussian noise.

In this thesis we first propose a feedback probability sorting algorithm with adaptive thresholding that reduces the computational complexity of the IRCSDFA by over 96%, with very little performance degradation. A similar but simpler non-adaptive sorting scheme was proposed for 4-ary 2D ISI channels in Zhu et.al. CISS2009, but its performance degradation could not be fully assessed due to the very high computational complexity of the full 4-ary IRCSDFA.

This thesis we then propose a Squared Euclidean Distance (SED) based local search post-processing scheme that operates on the final Log-likelihood Ratio (LLR) estimates output by the IRCSDFA. The basic and advanced schemes exploit the spatial correlation of the low reliability LLRs to find clusters, and then perform a local SED search on each cluster to find the estimation. Experiments show that the SED search yields performance improvements of up to 0.4 dB with less than 12.5% increase in computational complexity. As error clustering is present in many 2D ISI equalization algorithms, it is likely that the proposed SED search scheme could improve the performance of other 2D ISI algorithms as well.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER ONE: INTRODUCTION

This chapter gives a brief introduction to the iterative row-column soft-decision feedback algorithm (IRCSDFA), including the channel model, the trellis generation, related researches, and important design details. Part of chapter one and two of this thesis have been published in [1].

1.1 Channel Model

We consider the detection of an $M \times N$ binary equiprobable two dimensional (2D) independent and identically distributed (i.i.d.) image $\mathbf{f}$ with elements $f(k,l) \in \{-1,1\}$ from received image $\mathbf{r}$ with elements (pixels)

$$r(m,n) = g(m,n) + w(m,n), \tag{1}$$

where $g(m, n)$ is the 2D convolution

$$g(m,n) = \sum_{k} \sum_{l} h(m-k, n-l) f(k,l). \tag{2}$$

In the above equations $h(k,l)$ are the elements of a finite impulse response 2D blurring mask $\mathbf{h}$, the $w(m,n)$ are zero mean i.i.d. Gaussian random variables (r.v.s), and the double sum is computed over the support of $h(m-k, n-l)$: $S(m,n)_{h-} = \{(k,l) : h(m-k, n-l) \neq 0\}$. The model in (1) and (2) applies, e.g., to 2D data storage systems, which suffer from 2D intersymbol interference (ISI) at high storage densities. Such systems are under active development by industry for next-generation optical disk storage (e.g., [2]), and holographic data storage (e.g., [3]).

1.2 Related Research

Direct maximum likelihood (ML) detection of $\mathbf{f}$ from $\mathbf{r}$ requires comparison of $\mathbf{r}$ with 2^{MN} candidate images, and is impractical for typical image dimensions of $M, N \geq 64$. The standard Wiener filtering solution is significantly inferior to ML detection, especially at high signal-to-noise ratios (SNRs) [4]. Hence, it is desirable to develop a low complexity 2D detection algorithm that closely approximates the performance of 2D ML detection. For one dimensional signal, the Viterbi algorithm (VA) provides an efficient method for ML detection of ISI-corrupted data [5]. But the VA does not generalize to two or higher dimensions. In fact, it is been shown in [6] that the 2D ML sequence detection problem is NP complete for certain fixed 2D ISI channels having the form of (1). Asymptotically tight union bounds on the performance of 2D ML detection are developed in [7].Also, a number of papers (e.g. [8], [9]) have proposed efficient methods for computing the channel capacity of 2D-ISI channels.

A number of 2D decision-feedback VAs (DFVAs) have been constructed, based on row-by-row raster scanning of the image (e.g. [10], [11]). These VAs are hard decision algorithms. More recent work uses iterative algorithms that exchange soft information between soft-in soft-out (SISO) decoders.

In [9], the 2D convolution is decomposed into two 1D computations, and an iterative algorithm exchanges soft information between 1D SISO detectors. In [12], the binary source image is coded with a low-density parity check (LDPC) error correcting code before transmission over a separable 2D-ISI channel. Separability is exploited to construct an iterative row-column algorithm in which a non-binary column SISO detector is followed by a binary row SISO detector, followed by an iteration of the LDPC decoder, etc. The LDPC's coding gain enables [12] to approach, within less than 1 dB, the bit error rate (BER) curve for the non-ISI channel. In [13], soft information is exchanged between maximum a posteriori (MAP) row and column detectors; this scheme avoids decision feedback by making decisions on multiple rows/columns, rather than one row/column at a time. An iterative row-column soft decision feedback (SDF) algorithm (IRCSDFA) similar to that of [13], which outperforms the algorithms of [9] and [13] was also recently developed [14]. On the 2×2 averaging mask ISI channel, the IRCSDFA outperforms the separable algorithm of [12] (without LDPC coding) at high SNR by about 0.3 dB.

For variations of structures similar to IRCSDFA, [12] has a similar row-column scan structure, but algorithm [12] only apply for 2D channel mask that is separable to an inner product of two 1D masks, and examples in [12] only dealing with small size (3-by-3 and 5-by-5) masks. With the reduced complexity algorithm proposed in this thesis, should enable bigger separable masks to be equalized, and my research also shows that averaging masks and other classical masks can be separated, or at least with some approximation. To our knowledge, [15] has the best result by far in IRCSDFA with a novel iterative soft decision feedback zig-zag algorithm, within less than 0.1 dB of the maximum-likelihood performance bound for 2x2 and 3x3 masks.

1.3 Brief Summary of IRCSDFA

This thesis uses the iterative row-column soft-decision feedback algorithm (IRCSDFA) of [14] as a starting point; comparisons in [14] show that the IRCSDFA is among the best performing 2D ISI equalization algorithms published to date. The IRCSDFA's state and trellis definitions are based on those in [4]. An earlier publication, [13], is similar to the IRCSDFA, but does not employ extrinsic information LLR weighting, which has been shown to substantially improve the IRCSDFA's performance.

The IRCSDFA exchanges weighted LLRs between row and column soft-in/soft-out (SISO) estimators. Each estimator runs a BCJR-like algorithm (see, e.g., [16]) that uses soft decision feedback in the form of LLRs from previously processed rows or columns. For the

3 ×3 mask row SISO, trellis states and inputs are defined in Fig. 1, where Ω_1 and Ω_2 denote feedback rows. The algorithm structure of IRCSDFA is shown in Fig.2.

The 2D convolution in (2) can be viewed as the 2D inner product between original image $\mathbf{f}$ and the index-reversed and shifted mask $\mathbf{h}_{m,n}^{-}$with elements $h(m-k,n-l)$, where mask coefficient $h(0,0)$ is at pixel position (m,n). The inverted mask raster scans through the image row-by-row or column-by-column.

Trellis generation for the 3 × 3 mask on the mth image row is initiated by placing the input marked (m, n) in Fig. 1 at the left end of the row, where the initial values of the six state pixels are -1 due to the boundary conditions, and the vector $\mathbf{u}$ = $[u_{k0},\ u_{k1},\ u_{k2}]$ of three input pixels can take 2^3 = 8 different values. The entire state/input block is then shifted right to pick up the next three input pixels, and the previous three input pixels become the middle three state pixels. The full-state trellis therefore has 2^3 = 8 states with 8 branches out of each state. At each position (m, n), the trellis branch output vector consists of three 3 × 3 inner products between the index-reversed mask and the pixel values defined by the trellis. The upper inner product uses feedback from two previously processed rows, the middle uses one feedback row and the lower uses trellis pixels only. The branch metric is the SED between the branch output and the received pixel vector

$$\mathbf{y}_k = [y_{k0}, y_{k1}, y_{k2}] \triangleq [r(m,n), r(m+1,n), r(m+2,n)].$$

In the following description, we continue to assume the 3 × 3 mask definitions of Fig. 1. Given trellis state S_k, input vector $\mathbf{u}$, and received vector sequence $\mathbf{y}_1^{N_r} \triangleq [\mathbf{y}_1, \ldots, \mathbf{y}_{N_r}]$, define $\lambda_k^{\mathbf{i}}(s) \triangleq P(\mathbf{u}_k = \mathbf{i}, S_k = s, \mathbf{y}_1^{N_r})$, where $\mathbf{i} = [i_0, i_1, i_2]$ $\mathrm{N_r}$ is the finite length of recevied length, and $i_m \in \{-1,1\}$. We can then compute the *a posteriori* probability (APP) $P(\mathbf{u}_k = \mathbf{i} | \mathbf{y}_1^{N_r}) = \sum_s \lambda_k^{\mathbf{i}}(s) / P(\mathbf{y}_1^{N_r})$ using a modified BCJR algorithm. Specifically, the soft decision feedback (SDF) LLRs can be incorporated into the BCJR transition probabilities $\gamma_{\mathrm{i}}(\mathbf{y}_k, s', s)$, defined as $\gamma_{\mathrm{i}}(\mathbf{y}_k, s', s) \triangleq P(\mathbf{u}_k = \mathbf{i}, S_k = s, \mathbf{y}_k | S_{k-1} = s')$. The modified γ is the product of a modified conditional channel probability density function (PDF) $p'(\cdot)$, the trellis transition probabilities, and the extrinsic information from the other detector:

$$\gamma_{\mathrm{i}}(\mathbf{y}_k, s', s) = p'(\mathbf{y}_k | \mathbf{u}_k = \mathbf{i}, s, s')\, P(\mathbf{u}_k = \mathbf{i} | s, s^{'})\, P(s|s') \ \times P(\widetilde{\mathbf{u}_k} | \mathbf{u}_k = \mathbf{i}).$$

The factor $P(\widetilde{\mathbf{u}_k} | \mathbf{u}_k = \mathbf{i})$ is computed from the detector's extrinsic input LLR with the standard exponential formula; see, e.g., equation (2) of [14].

Denote the upper, middle, and lower inner products in Fig. 1 by $\mathrm{IP}_1(\Omega_1, \Omega_2)$, $\mathrm{IP}_2(\Omega_2)$, and IP_3, where we have explicitly indicated the dependence of the inner products on the feedback rows Ω_1 and Ω_2. The modified channel PDF is computed as an expectation over the values of inner products associated with state transitions $s' \longrightarrow s$ that are affected by past decisions:

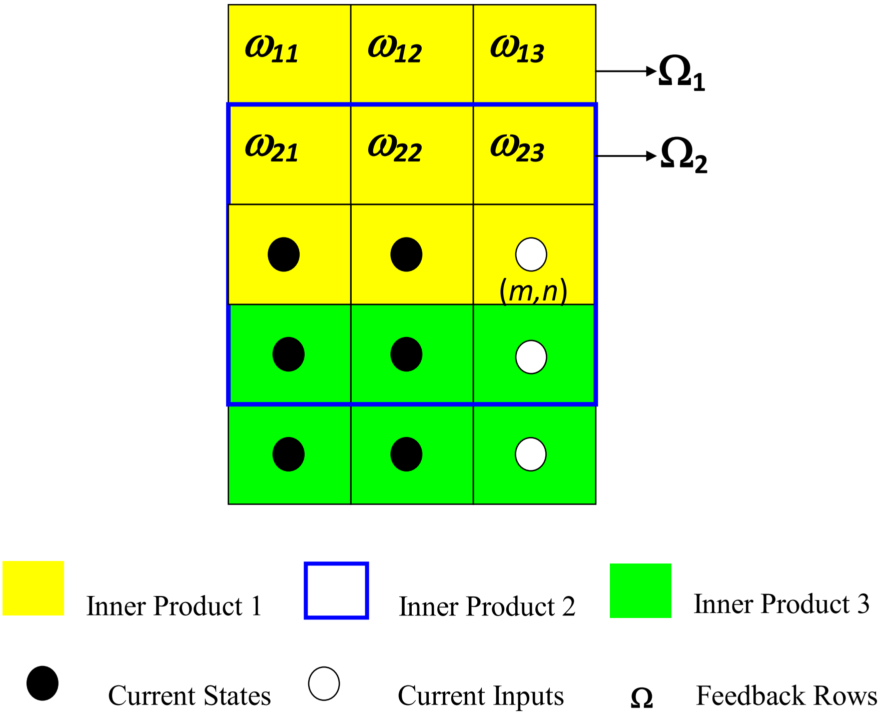


Fig. 1 State, input and feedback pixels for the 3 × 3 mask

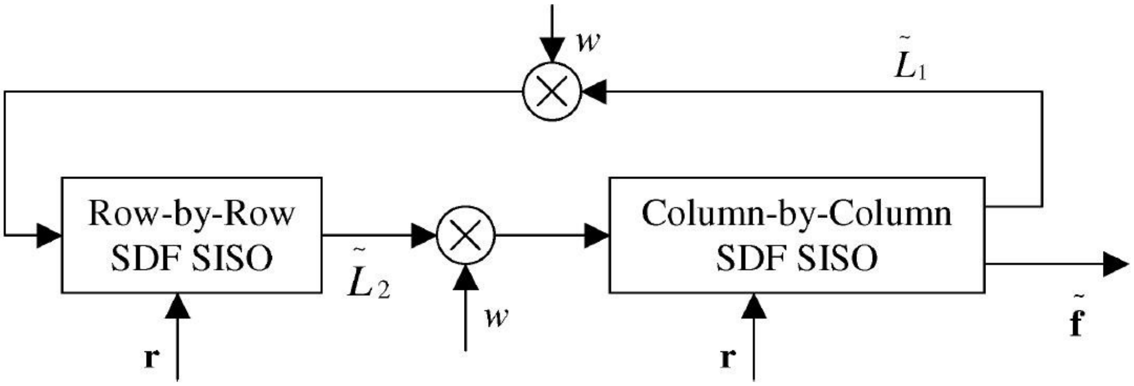


Fig. 2 Algorithm structure diagram of IRCSDFA

$$p'(\mathbf{y}_k|\mathbf{u}_k = \mathbf{i}, S_k = s, S_{k-1} = s')$$

$$= p(y_{k2}|u_{k0}, u_{k1}, u_{k2}, s, s') \times \left[\sum_{\Omega_2} P(\Omega_2) p(y_{k1}|u_{k0}, u_{k1}, s, s', \mathrm{IP}_2(\Omega_2)) \right.$$

$$\left. \times \sum_{\Omega_1} P(\Omega_1) p\big(y_{k0}|u_{k0}, s, s', \mathrm{IP}_1(\Omega_1, \Omega_2)\big)\right]. \quad (3)$$

In (3), the row probabilities are computed as:

$$P\big(\Omega_j = [\omega_{j1}, \omega_{j2}, \omega_{j3}]\big) = \prod_{l=1}^{3} P\big(\omega_{jl}\big)$$

where we assume that the feedback pixel probabilities $P(\omega_{jl})$ are independent. The $P(\omega_{jl})$ are computed from feedback LLRs from previously processed rows (or columns) during the current iteration. Since the received image is subject to additive white Gaussian noise (AWGN), the conditional PDFs of y_{k0}, y_{k1}, and y_{k2} in (3) are Gaussian with means $\mathrm{IP}_1(\Omega_1, \Omega_2)$, $\mathrm{IP}_2(\Omega_2)$, and IP_3.

To estimate the pixel located on (*m,n*) from the vector λs, we marginalize the λs over the pixels at (*m*+1,*n*) and (*m*+2,*n*):

$$\lambda_k^{i_0}(s) = \sum_{i_1,i_2} \lambda_k^{\mathbf{i}}(s). \tag{4}$$

The pixel LLR is computed as

$$L(k) = \log\left(\frac{\sum_s \lambda_k^{i_0=+1}(s)}{\sum_s \lambda_k^{i_0=-1}(s)}\right). \tag{5}$$

If $L(k) > 0$ pixel (m,n) is detected as +1; otherwise it is detected as -1, as in BPSK.

1.4 Notes on 2D Convolution and Boundaries

In Fig.2 we illustrate how the 2D convolution is defined in our previous code. The received value for a given pixel is computed by a 2D convolution of the pixels of the original page with the mask, and Fig.3 shows an example of 3-by-3 ISI mask.

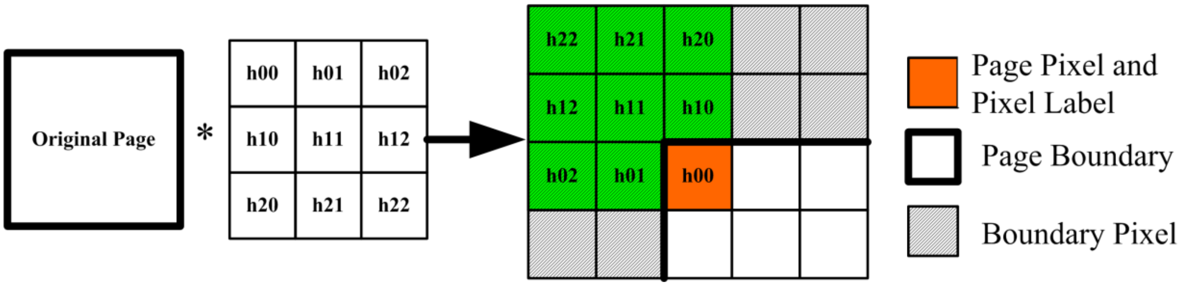


Fig. 3 Illustration of 2D convolution defined as in previous simulation code

When computing received pixels on the first/last row/column, extra pixels on the original page with known value is therefore needed. These extra rows and columns of pixels is boundaries, and we set their value to -1, as shown as shaded in Fig.3.

1.5 Full Termination of Trellis: Zero Expansion

In order to fully terminate the trellis generated in a row/column scan, we need the state on termination to be all zero. By the BCJR definition for a 3x3 mask, we need to step out three pixels of the far end of the boundary of the image (Zero Expansion), so that all six state pixels are boundaries with known value. But in our previous version the trellis stops by the end of the trellis, causing a mismatch of the α and β recursive computing. But this only causes trivial performance difference despite of SNR=14 as shown in Fig.4 (This comparison is done with all other configurations the same.)

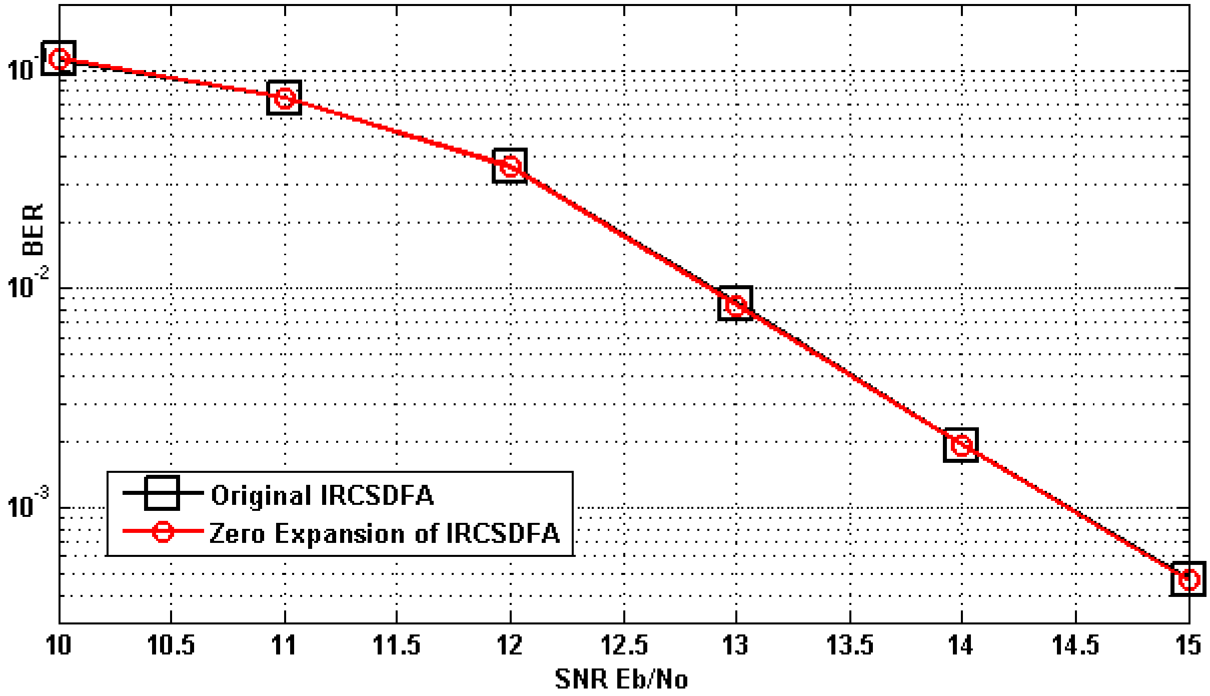


Fig. 4 Comparison of the original with the fully terminated (zero expansion)

1.6 Notes on Feedback pixel/vector in Row-Column Scan

In Fig.5 illustrates how we introduce the feedback pixels and feedback vectors. In order to calculate the convolution or inner production based on previous definition for the three input pixels in a row/column scan of a 3x3 mask, the extra rows/columns needed are feedback. These pixels' Log-Likelihood Ratio (LLR) is computed in the double sum in formula (3) of IRCSDFA.

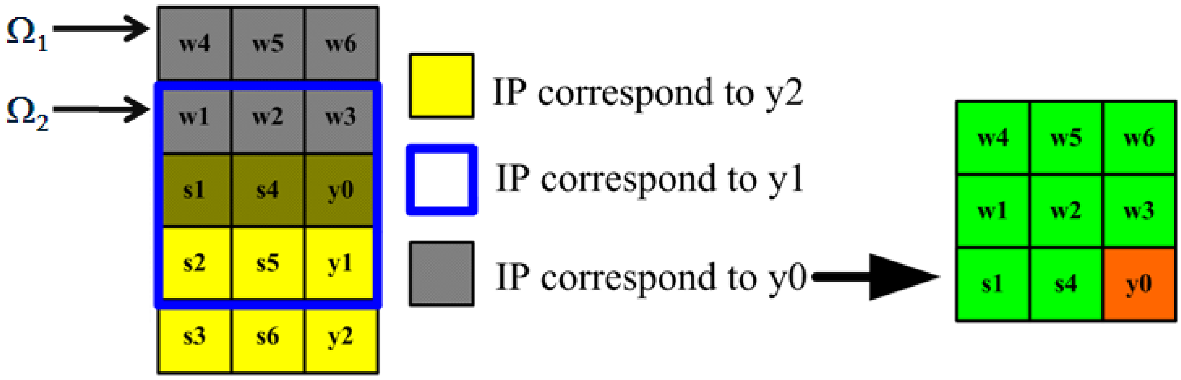


Fig. 5 Introduction of feedback pixel/vector due to the 2D convolution

1.7 Conclusion

In this chapter we presented the 2D ISI channel model, briefly summarized related researches in 2D ISI channel equalization, presented our channel model and trellis generation in IRCSDFA. In the following chapters we will present the work on complexity reduction and performance enhancement of IRCSDFA.

In the following comparisons, if not mentioned explicitly, all simulations are done on 128-by-128 randomly generated binary input pages and transmitted by BPSK, with enough boundaries set to -1 with 100 error count for stopping criteria, and the random seed is -52 for generating the noise sequence generated by the same random generator used by other researchers in the lab, using 3-by-3 averaging mask as the ISI channel and AGWN. We use bit error rate (BER) and SNR relationship in performance comparison, with the 100-error counts and 25% of further input pages as stopping criteria for the error accumulation simulation, to smooth out the variations due to the choice of random seed and random generator itself.

However, although we strictly go along with this 100-error count technique, there is still variation due to the different choice of random seed, as shown in Fig.6, and the variation is as large as 0.3dB and therefore can not be ignored. However, all experiments reported in this thesis can be reproduced exactly as it is, if using the same random seed, the same random generator, and the same way to use the noise sequence in the code, as also shown in Fig.6, where in two independent simulations with different structures but equivalently the same IRCSDFA parameter, achieve exactly the same performance.

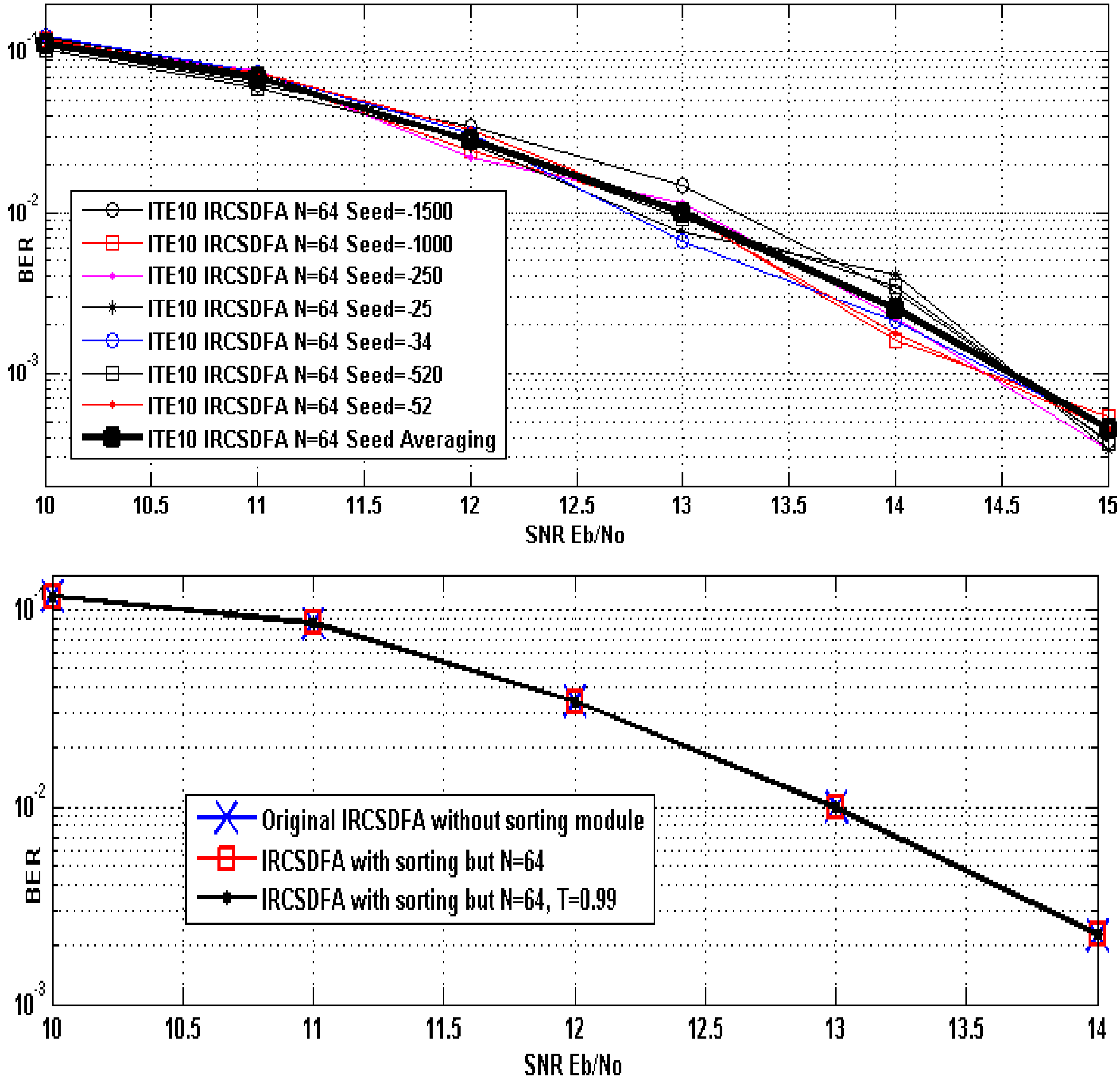


Fig. 6 Illustration on random seed variation and random seed in repeat simulations

CHAPTER TWO: FEEDBACK PROBABILITY SORTING FOR COMPLEXITY REDUCTION

In this chapter we introduce the feedback probability (FbPr) sorting algorithm for complexity reduction. First, the motivation of research is described, then two types of schemes for FbPr sorting: the basic scheme and the hybrid scheme are presented, with around 96% of complexity reduction, and generally equivalent performance compared to the IRCSDFA without sorting (the basic scheme), and over 20% of complexity reduction, with around 0.2dB in performance gain (the advanced scheme).

2.1 Motivation for FbPr Sorting

The expectation in (3) is taken over the 64 possible configurations of the feedback pixel rows Ω_1 and Ω_2 shown in Fig. 1. This expectation is computed for every branch in the trellis, making it a bottleneck constraining the algorithm's execution speed. To reduce the computational complexity of (3), it is important to note that the configuration probabilities $P(\Omega_1, \Omega_2) = \prod_{j=1}^{2} \prod_{l=1}^{3} P(\omega_{jl})$ of the six feedback pixels are highly non-uniform, such that only a few configurations contain most of the probability. This effect is illustrated in Fig. 7, which shows the cumulative probability distribution of the feedback pixels as a function of the number of configurations, for four iterations of IRCSDFA, averaging over all pixels of a 128-by128 random binary input image transmitted over the 3 × 3 averaging mask 2D ISI channel (where all nine mask coefficients are 1/9) at SNR=15dB. It is clear from Fig. 7 that the concentration of probability in just a few configurations increases with the number of iterations of the row-column algorithm. In practice, we use six iterations between row and column SISOs of the IRCSDFA, leading to even higher probability concentrations that is shown in Fig. 7.

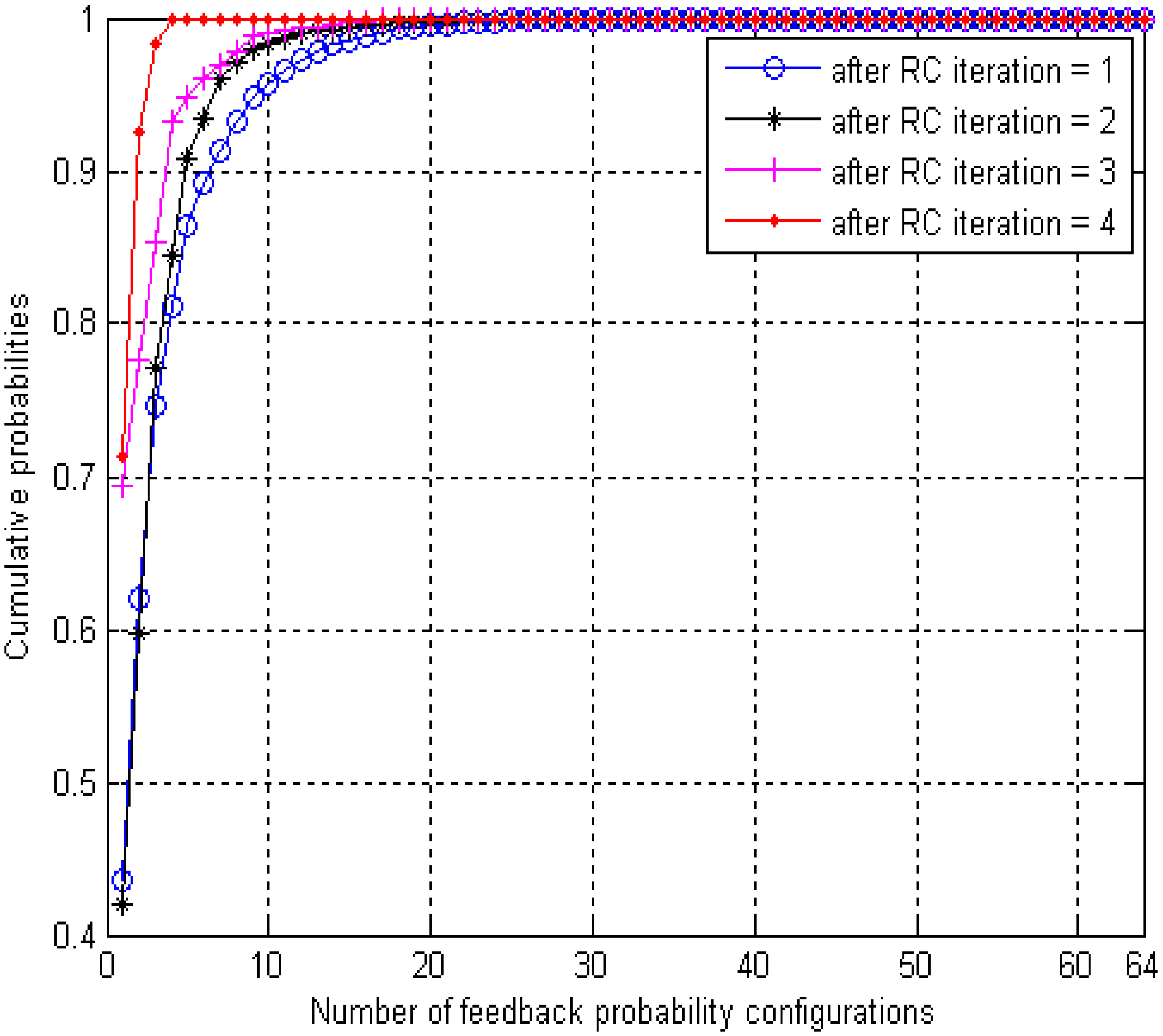


Fig. 7 Cumulative probabilities of the feedback pixel configurations

This observation implies that the computational complexity of the basic algorithm can be greatly reduced. The key idea behind our complexity reduction scheme is to retain only the high probability feedback configuration terms when computing the expectation in (3). Thus, before each computation of (3), we sort the 64 feedback configuration probabilities $P(\Omega_1, \Omega_2)$. Then we truncate the non-significant configurations.

2.2 Basic FbPr Sorting Parameter Set (*N*, *T*)

For the basic scheme we experimented with two truncation methods related to the basic parameter set (*N*, *T*). The first method keeps only the top *N* configurations; we refer to *N* as the "ranking decimator". The second method keeps the minimum number of configurations necessary to make the accumulated configuration probabilities exceed a percentage threshold *T* for each pixel.

Thus the fixed threshold *T* for a given pixel yields an adaptive ranking decimator *N* for accumulated configuration probabilities to exceed *T*, and different pixels therefore have different *N*. In Fig. 8 we show that a fixed threshold of *T* on the accumulated configuration probabilities for each position within one typical row of a 128-by-128 random input page yields an adaptive ranking decimator *N* for the pixel estimates on the row. In particular, some pixels have a higher ranking decimator *N* selected by the fixed threshold *T*. This suggests using a fixed *T* to perform a cumulative flatness check to see if the preset *N* is effective or not for a given pixel.

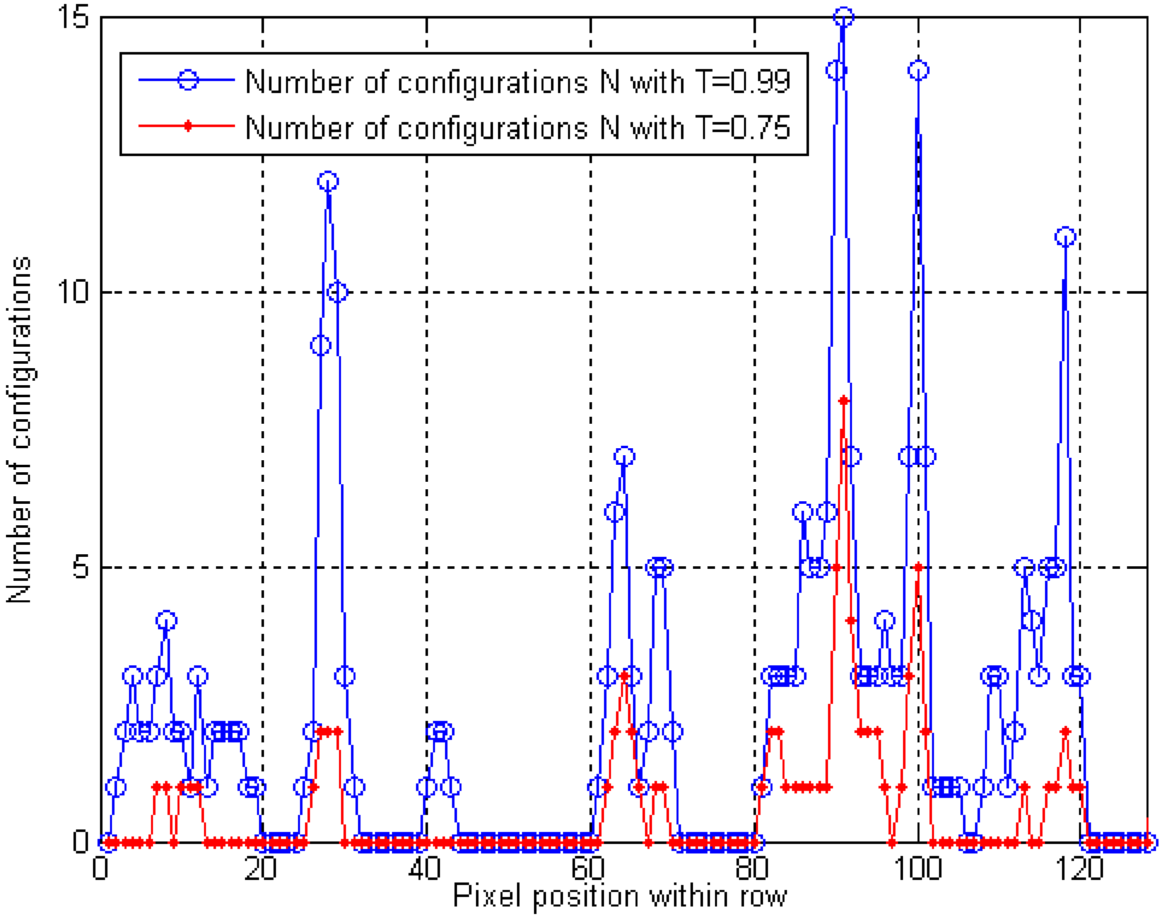


Fig. 8 Fixed *T* results in a different number of *N* to exceed *T* for a typical row

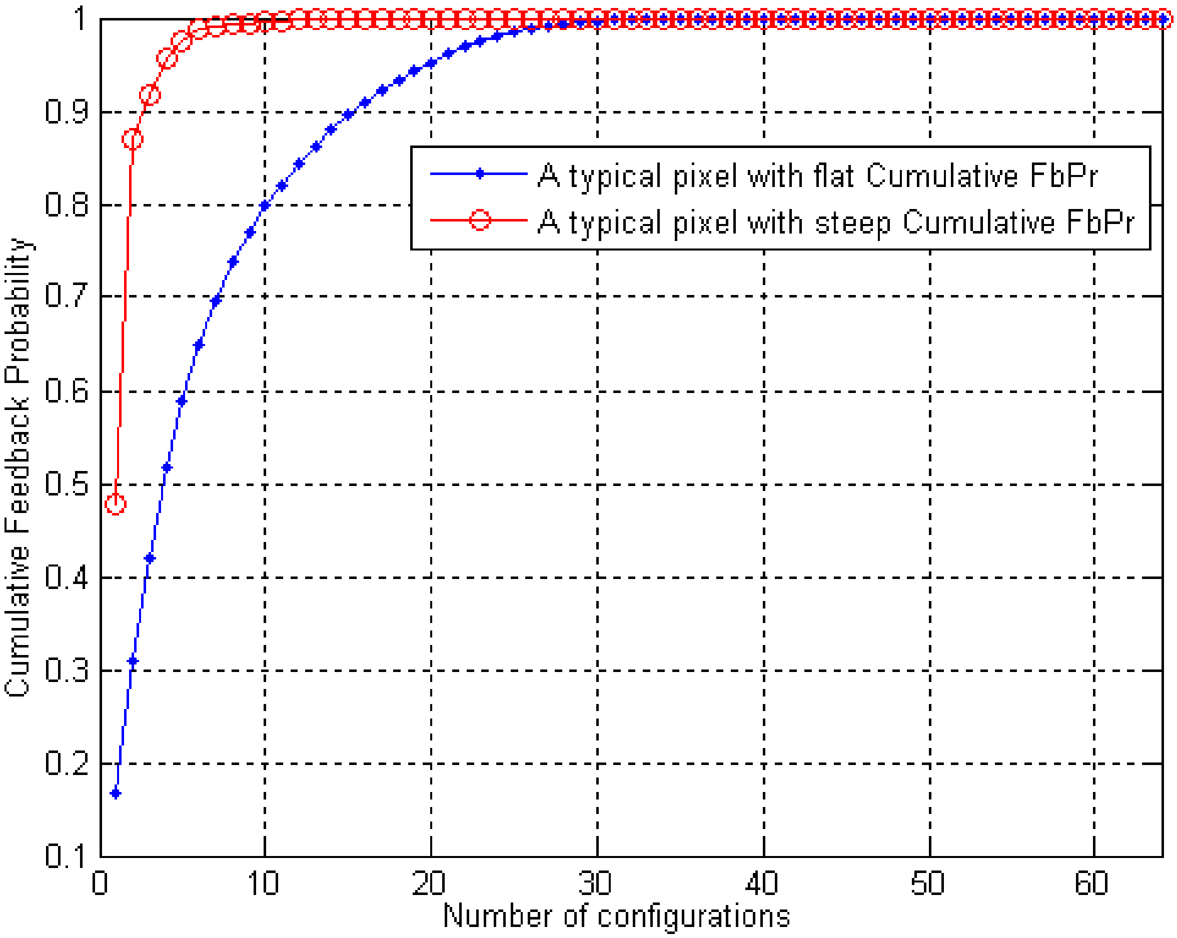


Fig. 9 Two typical pixels with steep and flat cumulative FbPr distributions

Fig. 9 shows two typical pixels with steep and flat cumulative feedback probability curves, further illustrating the need for an adaptive algorithm that can adjust N to achieve a desired T.

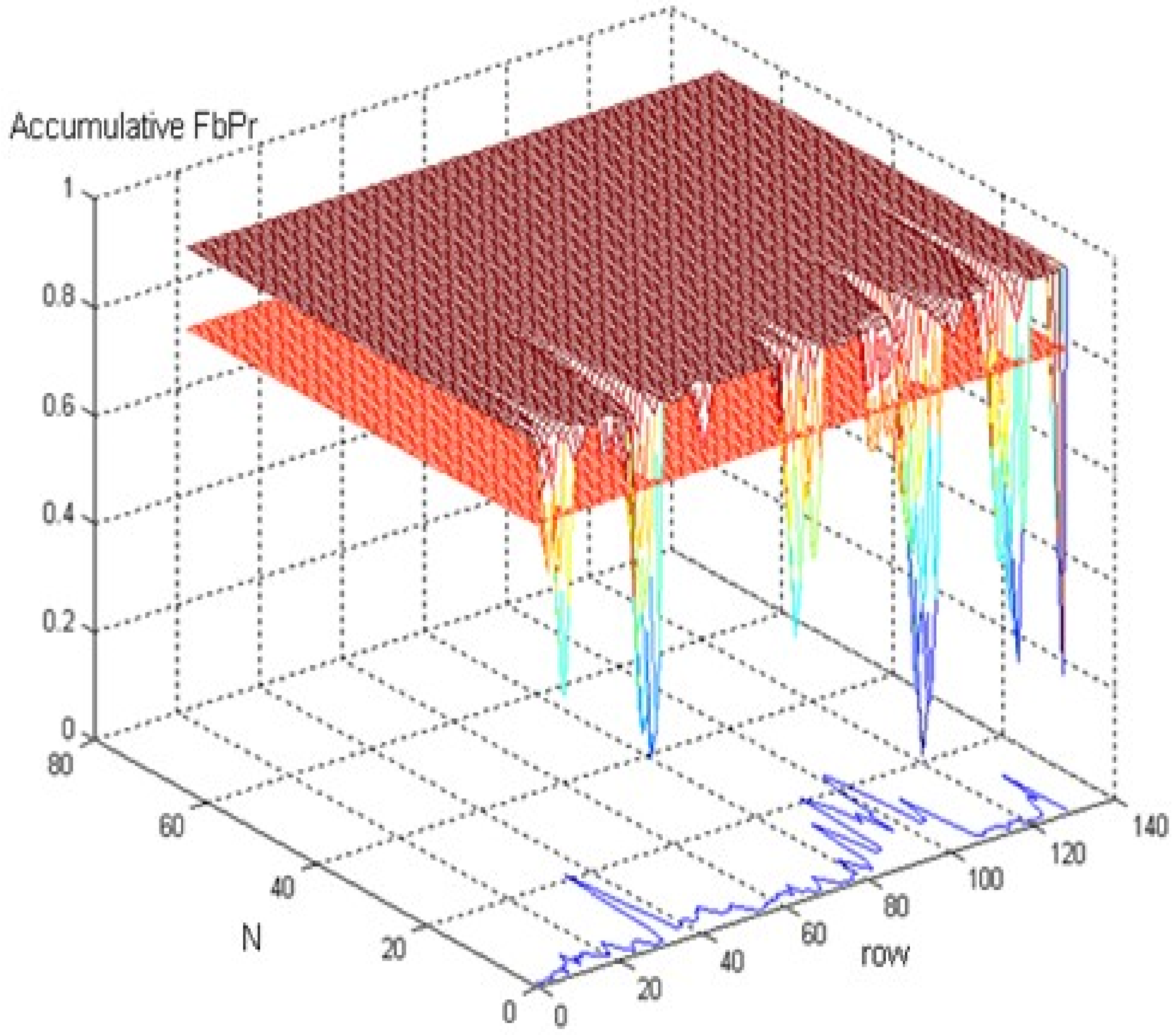
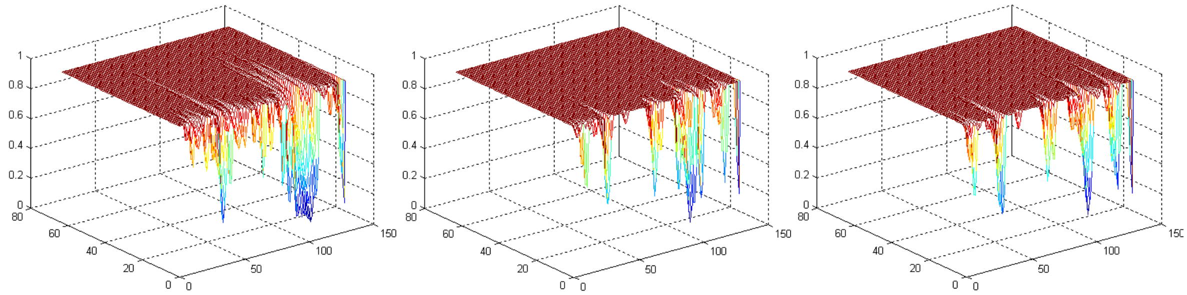


Fig. 10 Adaptive *N* by fixed *T* serves as a measurement for flat FbPr distribution



ITE=2 ITE=3 ITE=6

Fig. 11 Evolution of accumulative FbPr for a typical row along RC-BCJR iterations

Fig.10 further shows that fixed accumulative FbPr threshold *T* yields adaptive ranking decimator *N* and thus, pixels with flat FbPr distribution can therefore be detected. In Fig.10

the row axis is a typical row with 128 pixels on a 128-by-128 random image, the *N*-accumulative FbPr plane shows the accumulative FbPr for each pixel, and the top red plane represents a threshold of one; the middle orange plane shows the fixed threshold *T*=0.85; and the blue curves on *N*-rows plane shows the different adaptive *N* for each pixel to exceed that threshold *T*, showing the distribution of pixels with flat and steep FbPr in this typical row. Fig.11, a 3D version of Fig.10, shows the evolution of accumulative FbPr of a typical row with increasing iterations of RC-BCJR algorithm.

With sorting parameter (*N*,*T*) we have the structure for the basic sorting IRCSDFA as in Fig.12.

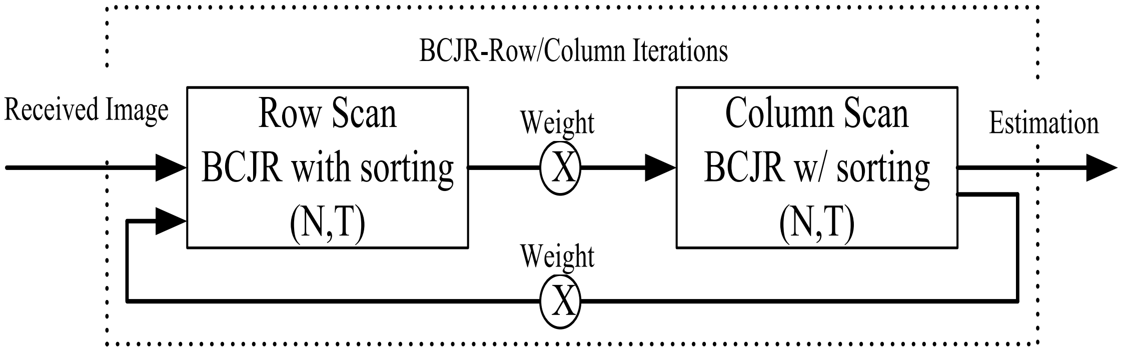


Fig. 12 Basic Sorting Structure for IRCSDFA

2.3 Performance of the Basic Schemes

Here we provide some performance comparisons with different parameter set. In Fig.13 we compare the performance only due to the ranking decimator *N*. We keep the top *N* biggest FbPr for each pixel along all BCJR iterations, and found out that with *N*=32, we have equivalent performance with half of the complexity reduced; and with *N*=2 we have a performance within 0.2dB of the original, with over 96% of the complexity reduced; but *N*=1 we have bad performance.

Note that this performance shows the only parameter change in the sorting. The performance of the sorting algorithm can be further enhanced by optimization of other IRCSDFA parameters.

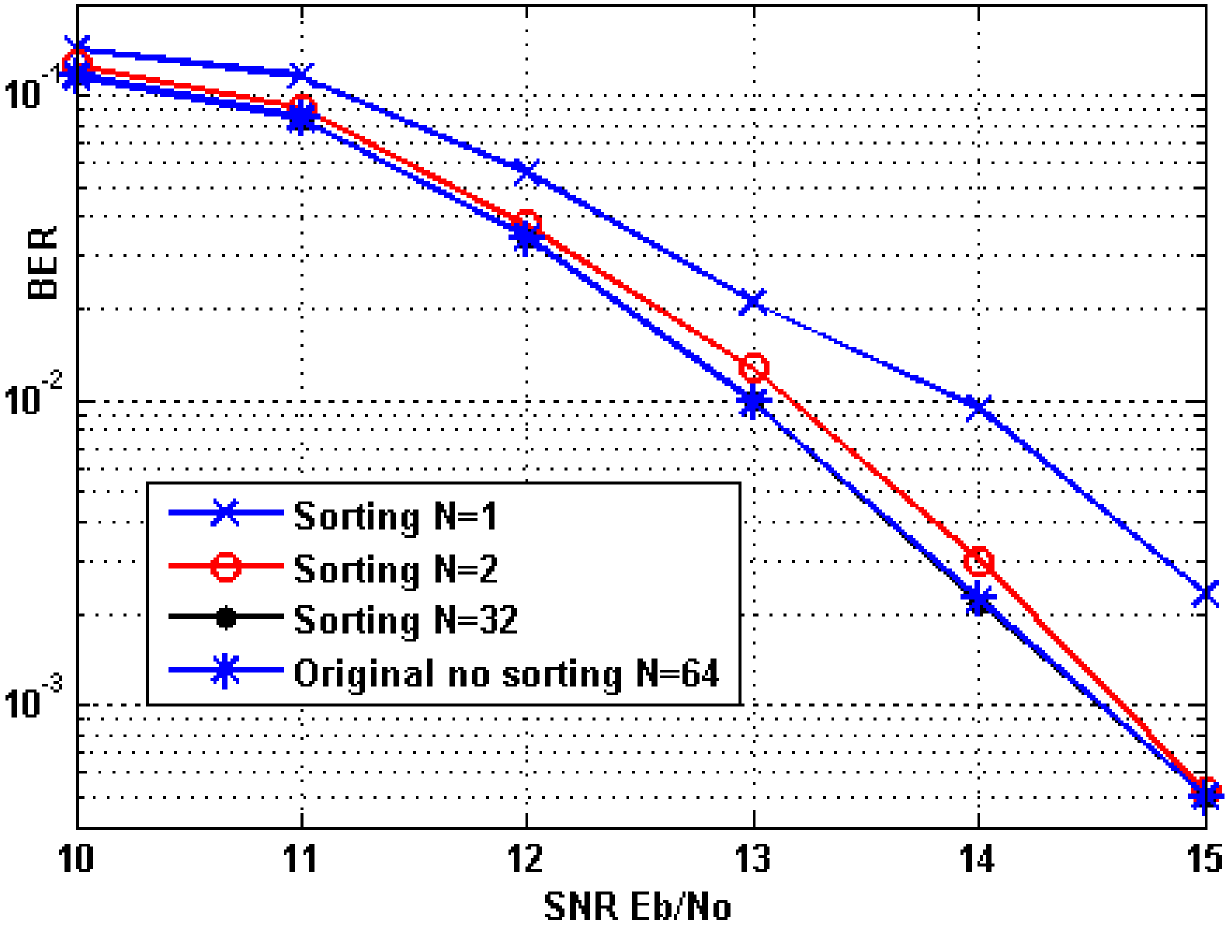


Fig. 13 Performance comparison only due to the choice of *N*

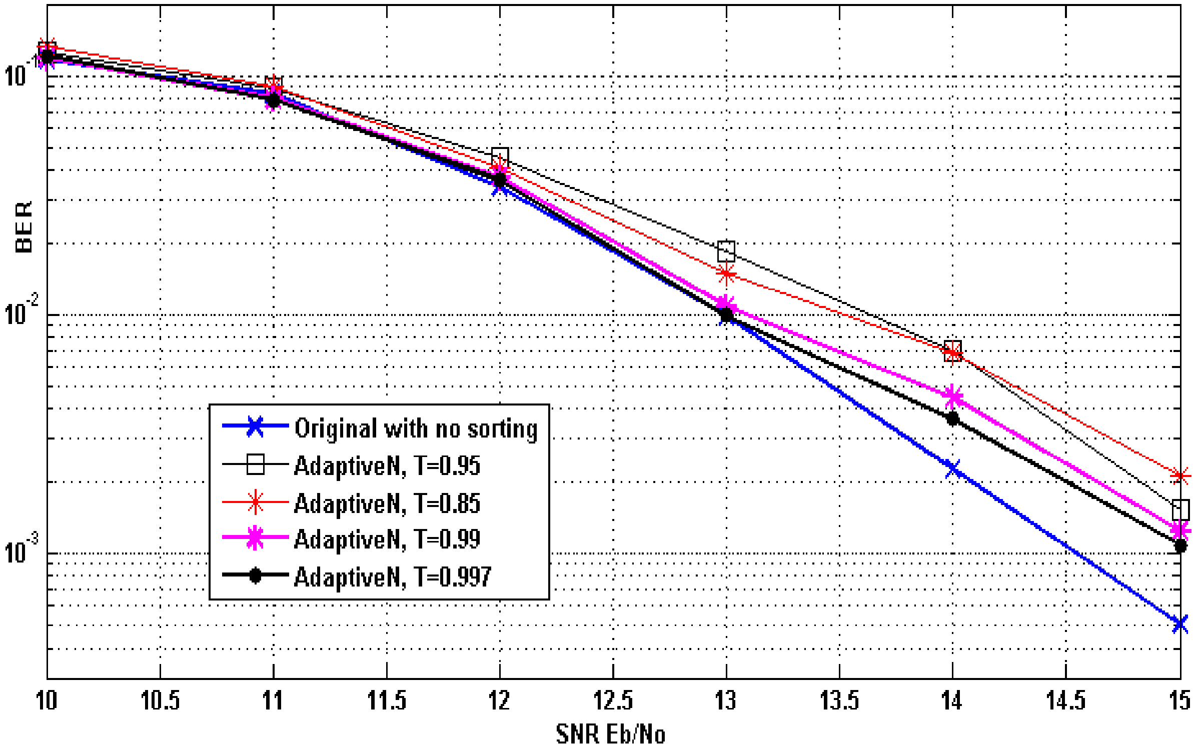


Fig. 14 Performance comparison only due to the choice of *T*

In Fig.14 we compare the performance for different value of the accumulative FbPr threshold *T*. We keep the adaptive *N* FbPr configurations for each pixel to exceed the threshold *T* along all BCJR iterations, and found out that on low SNR we have equivalent performances, but on high SNR we have bad performance. The reason is that if simply apply a fixed *T* for all pixels, the actual ranking decimator used for some pixel is as low as *N*=1, as in Fig.13, has bad performances. At SNR=15dB setting *T*=0.997 on one 128-by-128 random page, the *N* averaging on all pixels is as low as N_{avg} = 5.73 with 37.1% of all pixels actually used *N*=1.

By manually optimizing other parameters, we can further enhance performance as in Fig.13 and Fig.14 by using total 10 iterations instead of 6 used in row/column BCJR scan, using good quadratic weighting functions along all 10 iterations, and using zero expansion to fully terminate the trellis. The weighting function is for both 10 iterations and 6 iterations are in of the same order, but with different parameters so that the weight at final iteration is 1, as: $(1+\mathrm{k}^3)*0.008$ for a total iteration of 6, and $(1+\mathrm{k}^3)*0.0027$ for a total iteration of 10, with k stands for the iteration number.

Thus we got the best parameter so far, fix *N*=2 for all pixels and (*N*=2, *T*=0.99) means we use adaptive *N* due to threshold *T*=0.99 for all pixels, and if this adaptive *N* turns out to be smaller than 2, we set *N*=2. Here we use min *N*=2 as a threshold to automatically identify pixels with really steep FbPr distributions, and this gives us knowledge that too flat (slow convergence) and too steep (convergent too early) FbPr distribution are both bad. We denote this sorting strategy as (*N*=2, *T*=0.99)

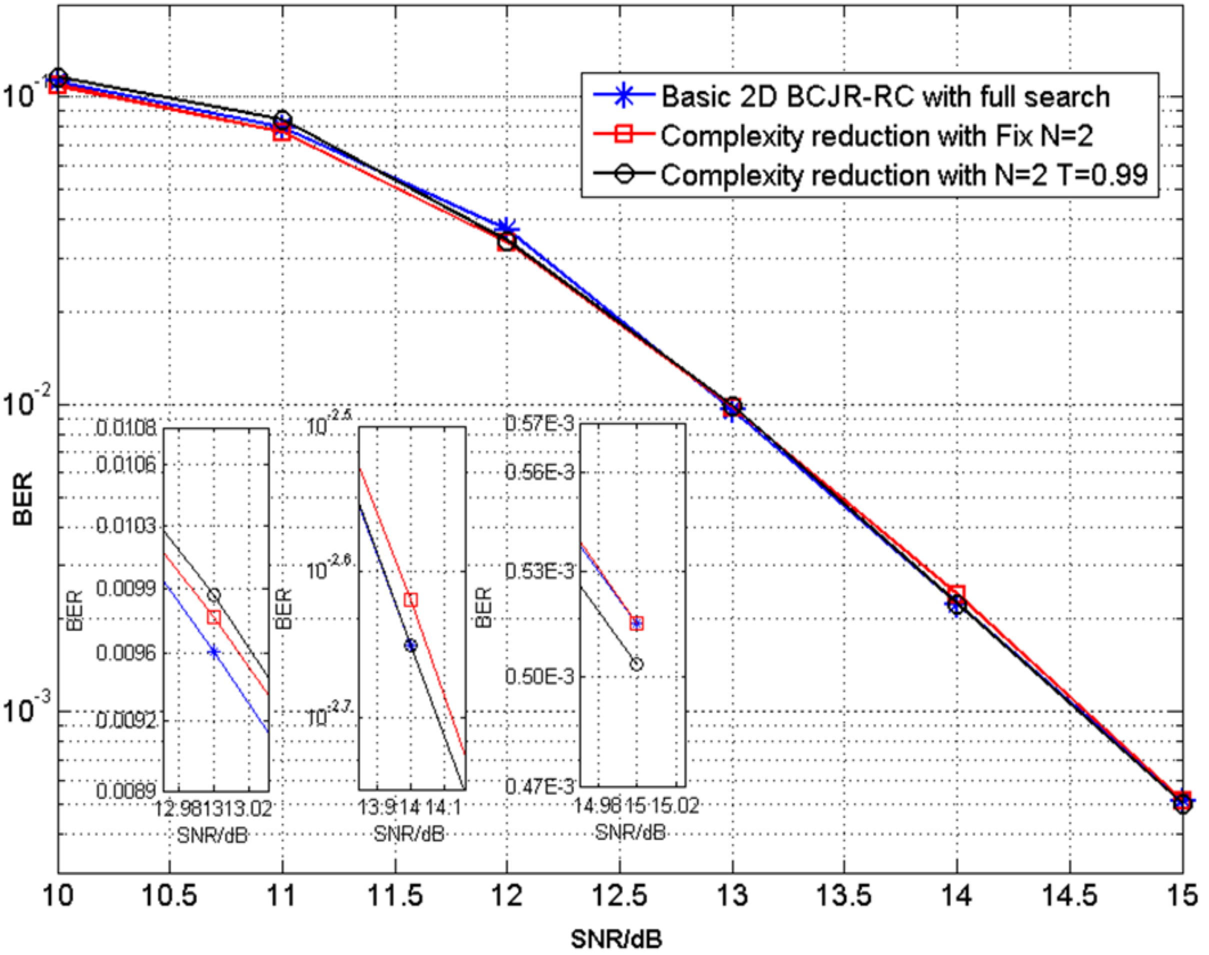


Fig. 15 Performance comparison of basic sorting schemes with best parameter set

In Fig. 15, it is clear that the proposed algorithm performs almost as well as the original algorithm. The adaptive ranking scheme with $N = 2$ and $T = 0.99$ gives a small but consistent gain over the scheme with fixed $N = 2$. The complexity reduction factor can be computed approximately as $N/64$, but an average value of N must be used for the adaptive scheme. If we fix $N = 2$ for all pixels, we reduce the complexity to 3.13% of the complexity of the original algorithm. For the adaptive case, an experiment with 50 128-by-128 random generated pages at a SNR of 15 dB showed that 12.67% of all pixels failed the T=0.99 cumulative probability check, and that the average ranking decimator N was 6.81 (compared to previous N_{avg} = 5.73 as in Fig.12). Thus the complexity of the adaptive case is approximately 5.43% of the complexity of the original. Both techniques offer a great amount of complexity reduction with almost no performance loss, as shown in Fig.16.

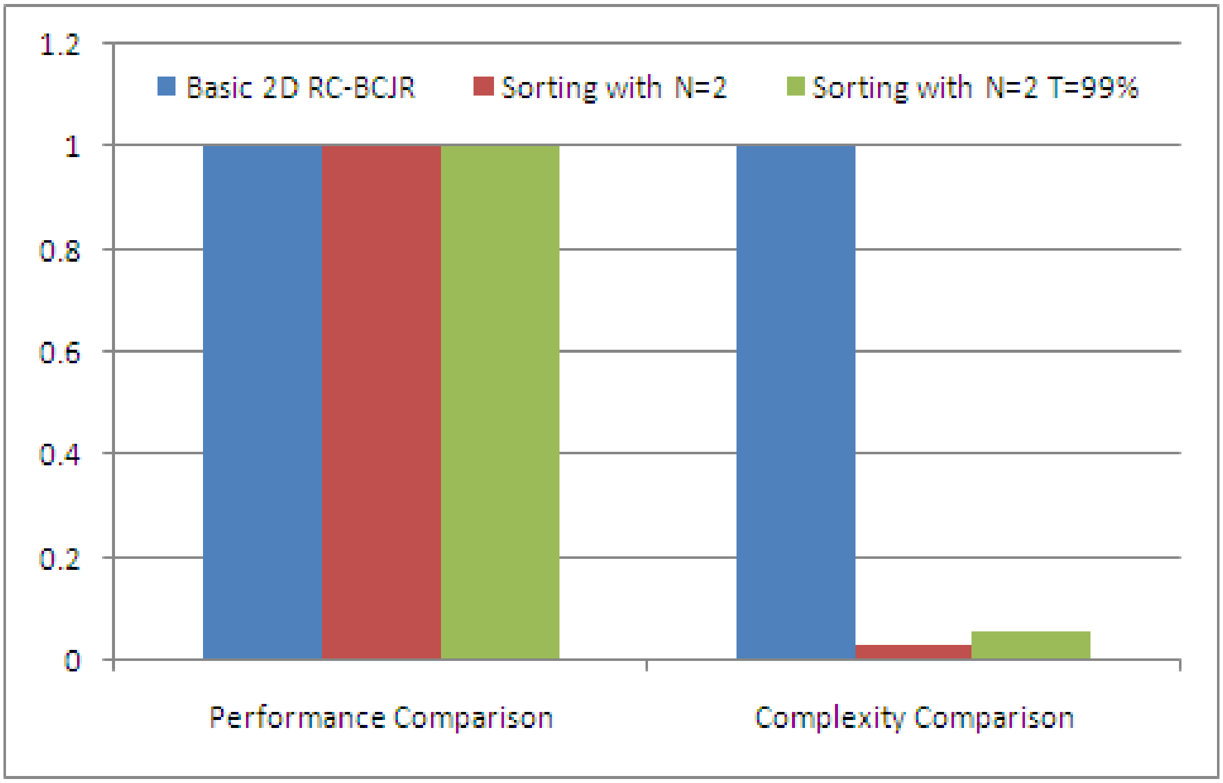


Fig. 16 Performance and complexity comparison for best basic sorting schemes

2.4 Advanced Schemes for Performance Enhancement

Beyond the performance enhancement achieved by the basic sorting schemes by manually optimizing the algorithm parameters, the motivation for advanced sorting scheme is to try to further enhance the IRCSDFA performance while keeping over 80% of complexity reduction, using hybrid parameter and hybrid algorithm structure such as those mentioned in subsections 2.5 and 2.6 below.

2.5 LLR Masking

From the discussion for Fig.13 and Fig.14 we know that converging too slow or too fast are both bad. Thus the general idea is that, we consider using some masking filtering to make a pixel's LLR merge with its neighbors, thus smoothing and slowing down the convergence of some pixels' LLRs if they were converging too fast.

There are two dimensions for judging the convergence speed: judging by neighboring pixels in the same iteration (x-y plane), and judging by history (evolution) of a certain pixel along different iterations (z axis), as shown in Fig.17.

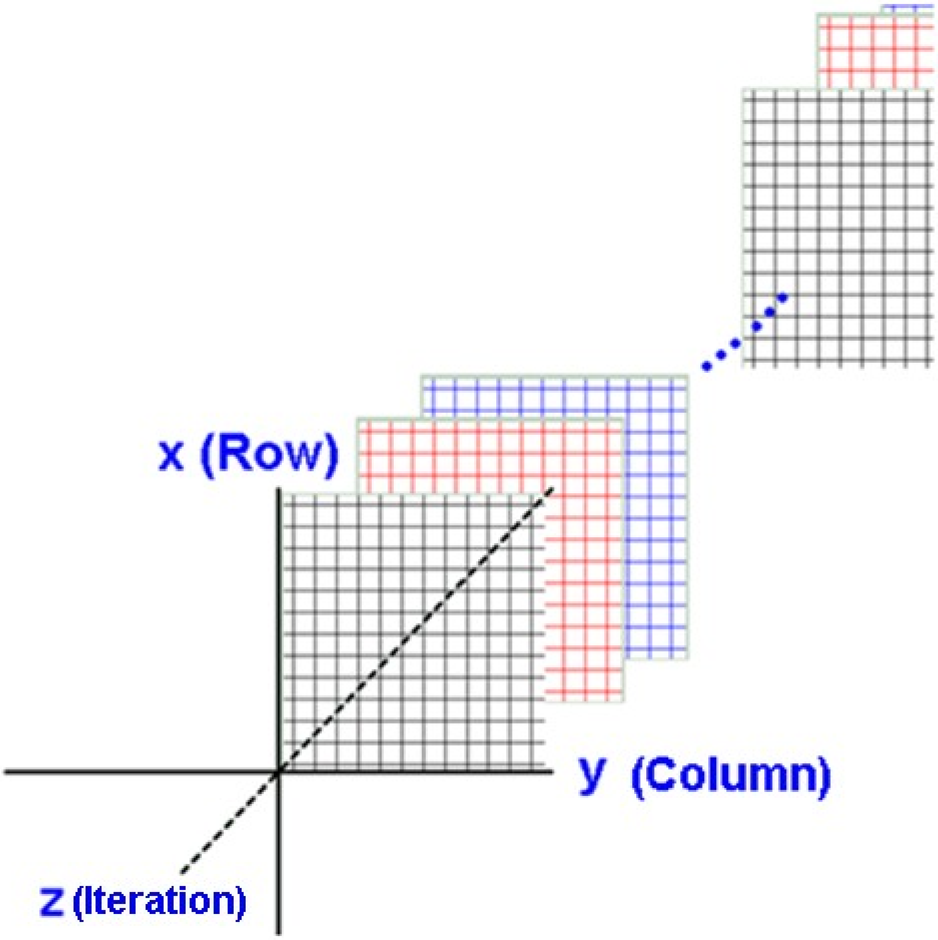


Fig. 17 Illustration of two dimensions for judging convergent speed

Observation of statistics over 10 randomly generated input pages show that there is no strong relation between the evolution of LLR values of a pixel and the event that the pixel is incorrectly estimated. Thus this LLR masking method judges a convergence speed by its neighbors. The structure of this LLR masking advanced scheme is shown in Fig.18.

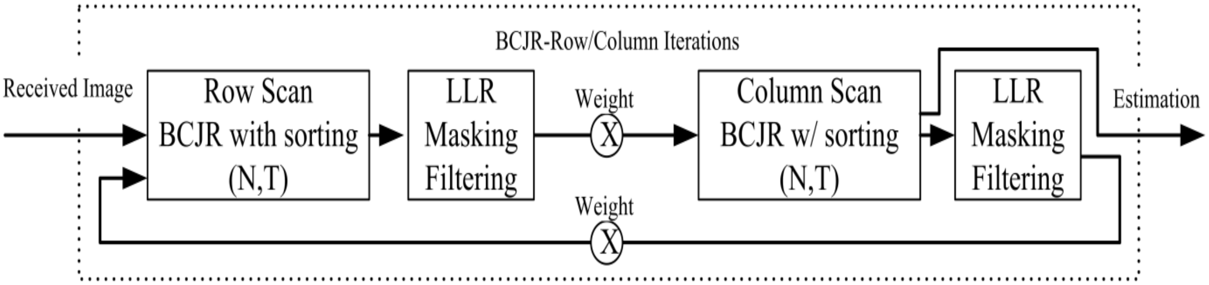


Fig. 18 Structure of LLR masking for advanced scheme

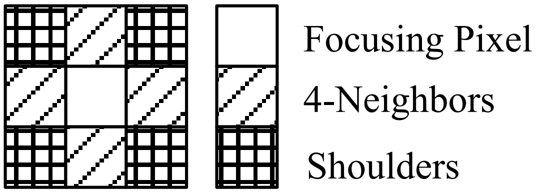


Fig. 19 Illustration of Pixel Topology

Several LLR designs were tried before finding one structure with good performance as shown in Fig.20. The pixel topology is defined in Fig.19: for a focusing pixel, its 4-neighbors and shoulder pixels are defined. Observations of statistics over 10 randomly generated input pages show that, nearly all of the wrong estimations have pixels with very low LLR magnitudes on their shoulders, and for those shoulder pixels, their 8-neighbors are mostly correct estimations, with higher LLR magnitudes. Therefore we use a LLR threshold to detect those low LLR magnitude pixels as set of pixels that are highly likely to be wrong estimations, and apply local 3-by-3 (or other size) averaging masking on their LLRs, and then use this result as the output LLR of the current BCJR MAP row/column scan.

Rules for this algorithm are:

(1) the pixels that have their LLR value changed in the masking are the low LLR magnitude pixels detected and their shoulder pixels; other page pixels near them are involved in the inner product by the definition of 2D convolution, but their LLRs remain the same;

(2) The value taking part in the convolution is the estimation probability computed from the LLR, and the changed probability is converted back to LLR.

(3) If in case the sign of the pixel is changed during this process, we make the decision by scanning its neighbors. If there are more pixels with low LLR magnitude in its 8-neighbors (more than 3), we admit the change of sign. If there are less than 3 but they are

all on its shoulders, we still admit the changed sign. If they are not all on its shoulders, we only admit the change of magnitude. If none of them are shoulders, we set that LLR to zero.

This algorithm is named PrNC33ShdSzα/β, describing its major characteristics such as: Convert LLRs into probabilities for masking, Neighbor Check; 3x3 Masking; Shoulder Scan and LLR Set Zero; α is the starting iteration number for the LLR masking filtering module, and β is the total number of iterations of the RC-BCJR MAP algorithm. For example 06/10 means for a total of 10 iterations, the masking is only applied starting from the 6th iteration.

Other algorithms compared in Fig.20 are similar. NC33α/β means we use LLR value for masking, also there is no Shoulder-Scan and Set LLR to Zero part, i.e. we either admit the change of sign, or keep the sign and only change the magnitude. NC33Abs α/β is similar to NC33α/β, except that, we ignore the sign during the LLR masking, and there is no change of LLR signs.

In Fig.20 we only present these three structures which have relatively large performance enhancement. For many other variations tested, either they have small or marginal performance enhancement, or their performance crosses over with that of the original algorithm.

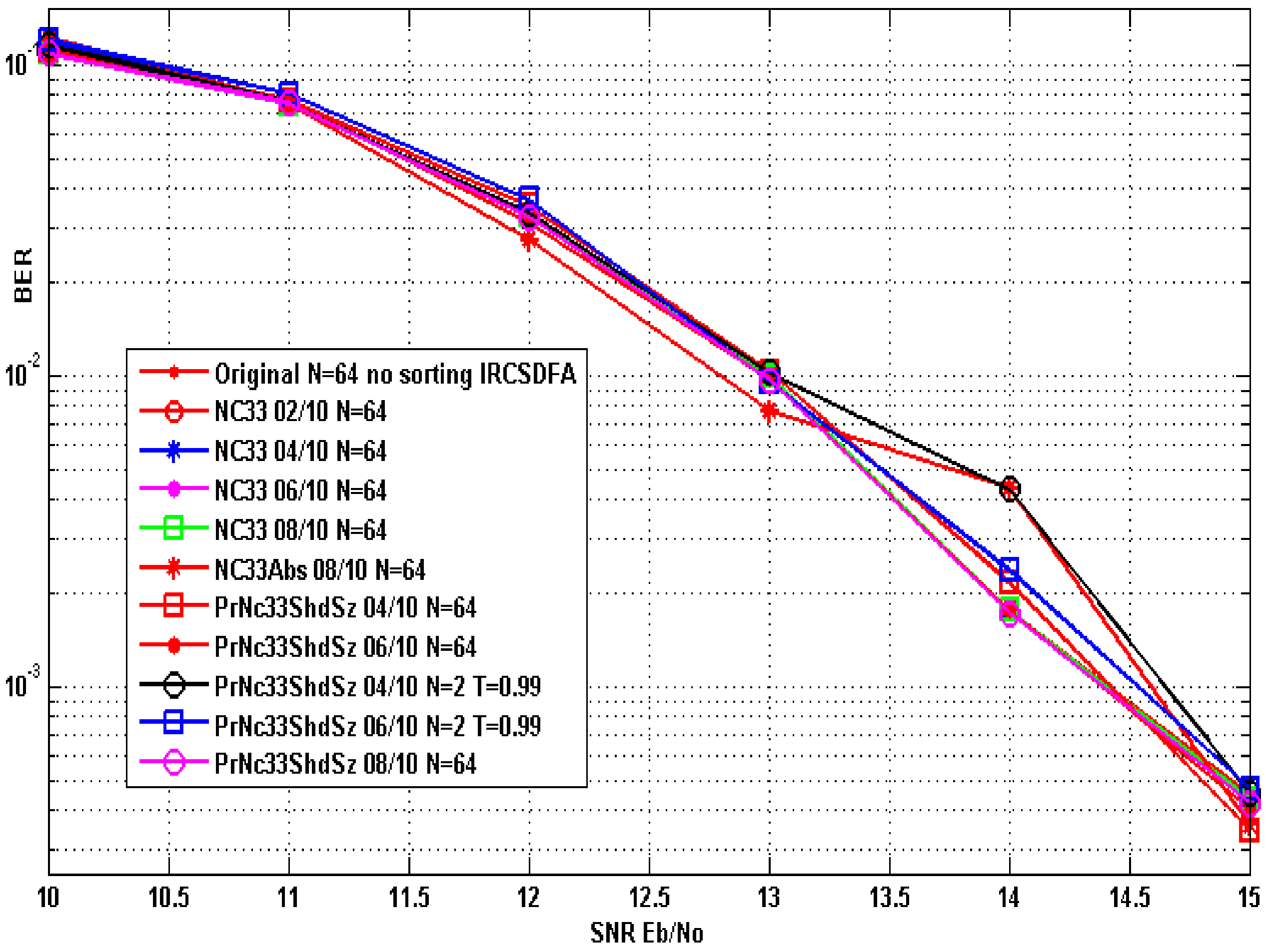


Fig. 20 Comparison of different LLR masking algorithms for advanced scheme

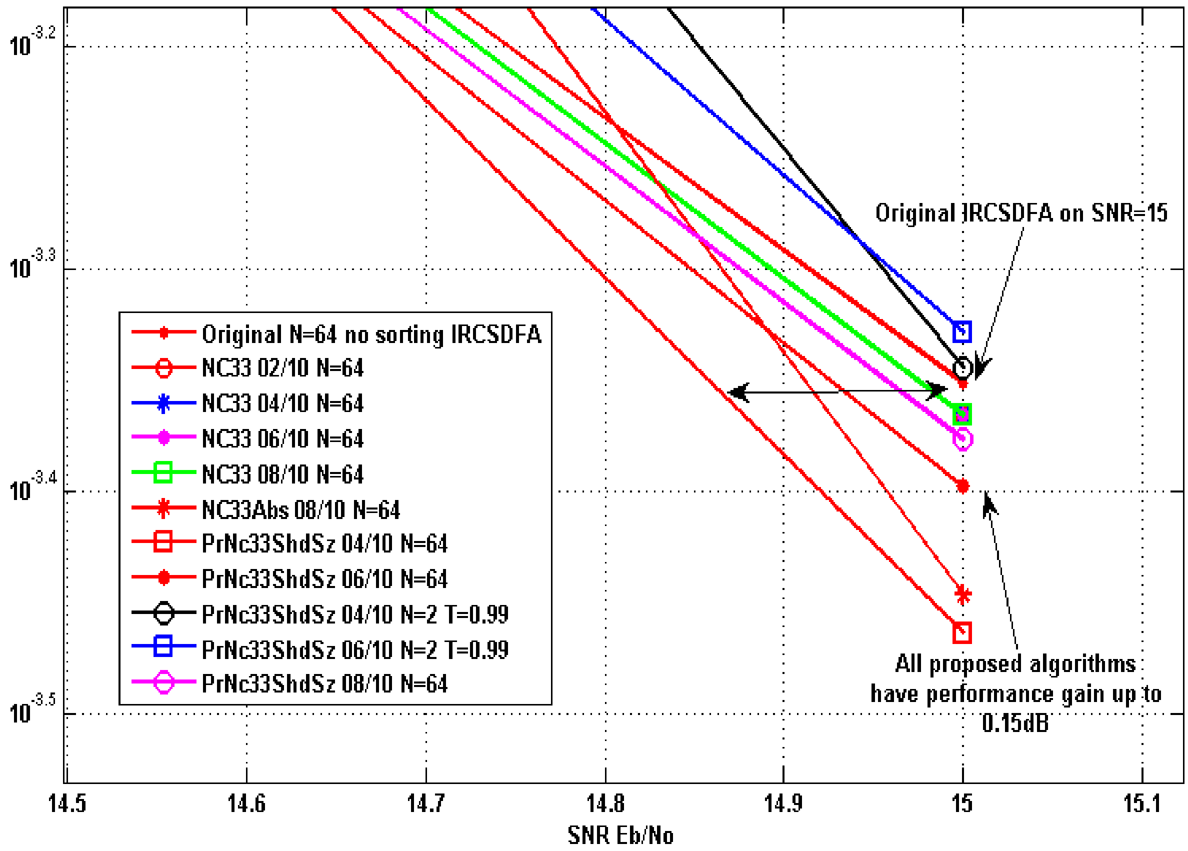


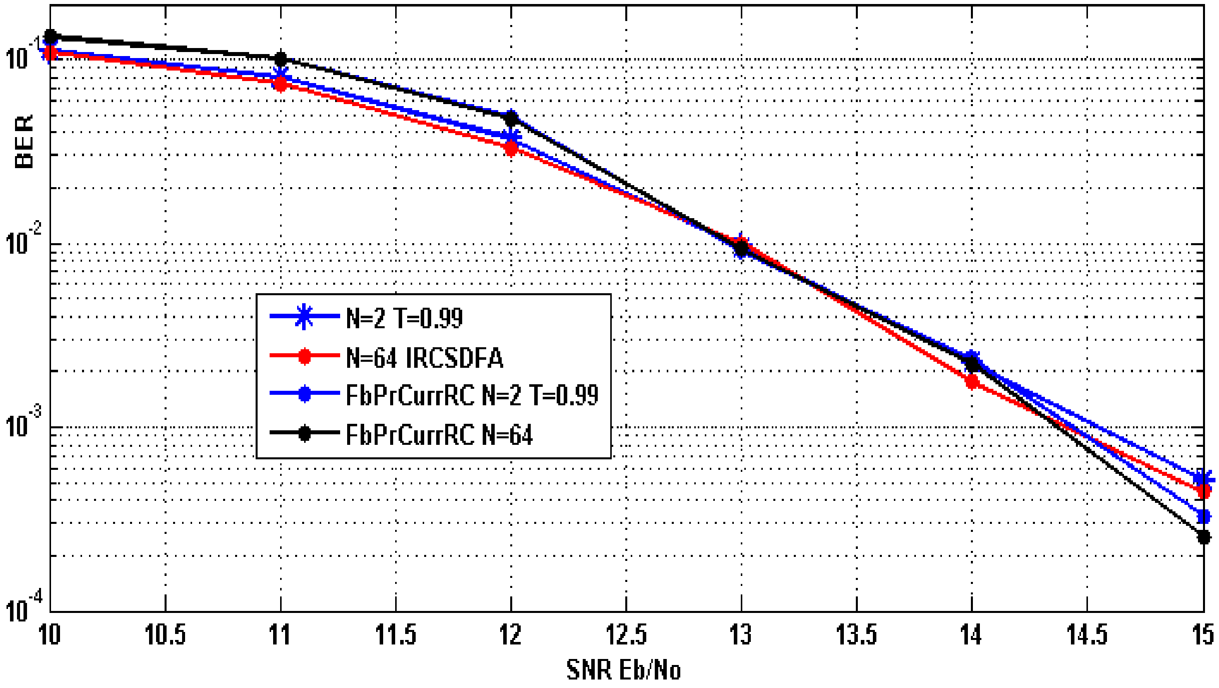
Fig. 21 Magnify at SNR=15dB showing performance gain

In Fig.20 we can see that, for SNRs less than 14dB, the performance is equivalent within a variation of 0.2dB. The magnified plot in Fig.20 shows that at SNR=15dB, three proposed algorithms without sorting have up to 0.15dB of performance gain compared to the original IRCSDFA, and their corresponding versions with sorting have competitive performance with the original IRCSDFA.

2.6 Parallel Structures

Several variations of parallel structured IRCSDFAs were investigated. Fig.23 shows one full iteration of the proposed parallel algorithm. Thus if the total number of iterations is the same as the original IRCSDFA, the complexity is doubled. Also note that the LLR merging is just a LLR plus, or probability multiply. The presented parallel structure has 6 such iterations, which is equivalent to 12 iterations of the original IRCSDFA. Also note that, due to the separated parallel structure, the algorithm can be easily adapted to a multi-core multi-task computing system. Other paralleled structures were also simulated, but the performance is not good.

Fig.22 shows there is some performance degradation at low SNR, but at high SNR=15dB, the parallel with sorting has 0.15dB gain over the non-sorting IRCSDFA with 19.3% of complexity reduction, and around 0.2dB gain over the sorting IRCSDFA but 1.2 times the complexity of the best basic schemes The parallel scheme without sorting has about 0.25dB gain over the non-sorting original IRCSDFA with 1.2 times the complexity.



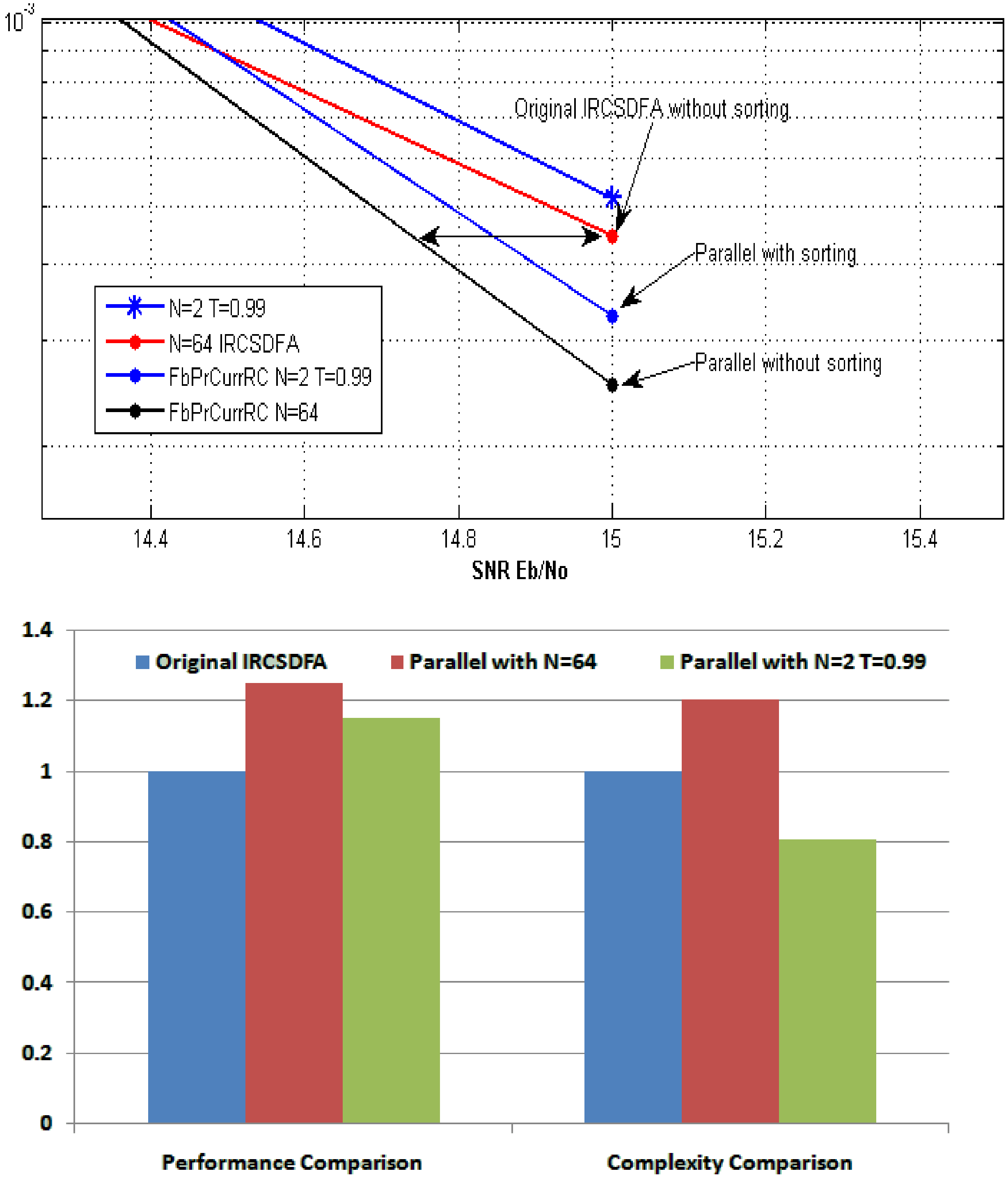


Fig. 22 Performance comparison of parallel structure for advanced scheme

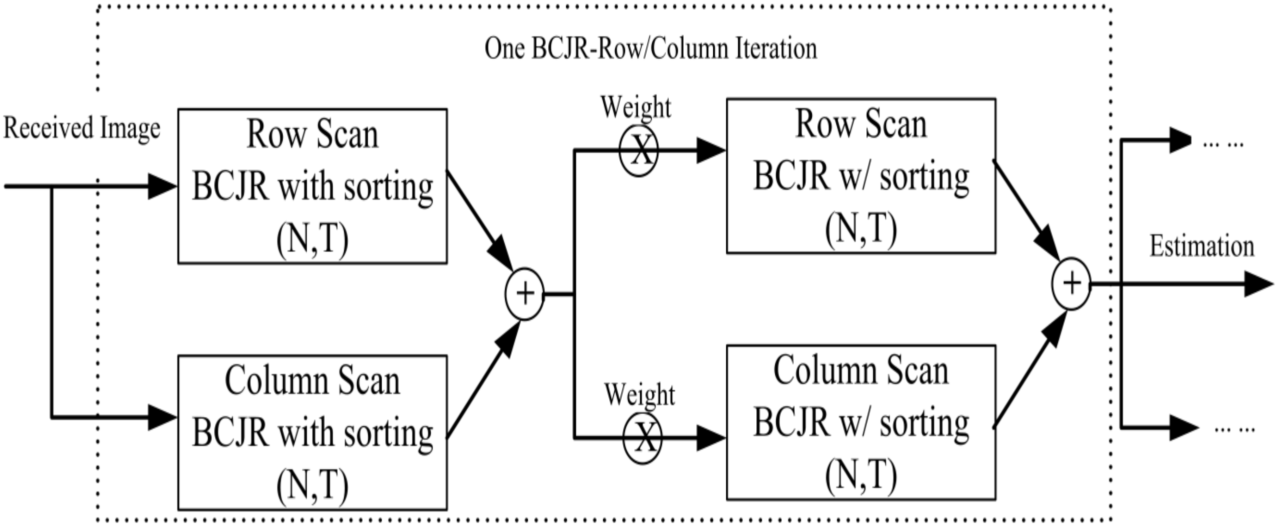


Fig. 23 Structure of the presented parallel structure for advanced scheme

2.7 Conclusion

In this chapter we presented the FbPr sorting algorithm for the IRCSDFA. We show that the basic scheme with parameters (*N*=2, *T*=0.99) has over 96% complexity reduction and equivalent performance (or even a little bit better) compared to the original IRCSDFA. We also show two advanced schemes with and without sorting, for up to 0.25dB of performance enhancement and also some complexity reduction.

CHAPTER THREE: SED BASED LOCAL SEARCH FOR PERFORMANCE ENHANCEMENT

In this chapter we introduce the Squared Euclidean Distance (SED) based local search as a LLR-based post processing of IRCSDFA for performance enhancement. First, the theory of SED local search is introduced, then the basic scheme and the advanced scheme is given. We give details about both schemes by define the terms and notations used, describing the step-by-step procedure we followed, and the performance. This SED based local search algorithm can enhance IRCSDFA by up to 0.4dB in performance, with modest additional complexity required. Also note that our SED local search module can also be applied to other 2D ISI channel equalization with error clustering characteristic for performance enhancement purpose.

3.1 Introduction to SED based local search

For the channel as modeled by (1) and (2), the full ML search procedure for an $M \times N$ binary input image involves convolving all 2^{MN} candidate source images with the ISI mask, and choosing the estimated source image $\hat{\mathbf{f}}$ as $\hat{\mathbf{f}} = \text{argmin}\left(\text{SED}\left(\mathbf{r}, \mathbf{h} * \hat{\mathbf{f}}\right)\right)$, where $\mathbf{r}$ is the received image. While this procedure is clearly prohibitively complex, we show in the following that local "Maximum Likelihood (ML)" searches over suitably chosen clusters of low reliability IRCSDFA output pixels can improve performance at modest complexity cost.

The reason we get interested in and the motivation for this SED based local search on selected clusters of pixels, or local ML search is that, the full ML search, or Viterbi Algorithm, efficiently performs a global search and has been proved to have near-bound performance in 1D ISI equalization. And thus we hope that local ML search can further enhance the performance of IRCSDFA.

3.2 Basic Scheme for SED local search

First we define the terms and notations used in the basic scheme:

TE: pixels that are true estimation errors

Least reliable set, "the set": set of pixels with LLR magnitude lower than a given and fixed threshold ***T***. Based on the definition of LLR, lower magnitude means less probability in estimation and vice versa, higher probability in wrong estimation. Therefore those pixels are those we are interested in performance enhancement.

The threshold *T* is a threshold on the probability computed out of LLR, e.g. the T59 denotes probability *T*=0.99999=1-1e-5, equivalent to LLR magnitude threshold of 11.513. All pixels with estimation probability computed out of their LLR less than *T*=0.99999 or LLR magnitude less than 11.513 are included in the set; and similarily T89 denotes probability threshold *T*=0.99999999=1-1e-8, equivalent to LLR magnitude threshold of 18.421.

Clusters: statistics show that the TEs, the least reliable set pixels, tend to cluster together along the iterations of RC-BCJR algorithm, as shown in Fig.25. In order to make use of this spatial correlation, we scan all the set pixels and by 8-neighboring connectivity check, divide those into clusters. In the basic scheme, we simply ignore the single set pixels with no other set pixels 8-neighboring to it, or clusters with size equal to one is eliminated for further processing.

ISI Masking Region: the local search algorithm computes the SED for all possible combination of the value of the cluster pixel. Therefore for each possible combination of cluster pixels value, we need to perform the 2D ISI to simulate the channel inter symbol interference. By the definition of the 2D convolution in chapter one, in order to compute the received value in the position of cluster pixels, extra pixels needed according to the mask size. These extra pixels are the ISI masking region of the cluster.

SED local search region: The pixels involved in the SED based local search algorithm include the cluster pixels together with its ISI masking region; this set is defined as the SED local search region.

Figure 24 illustrates the steps involved from thresholding, getting the least reliable set, forming clusters and getting the SED local search region.

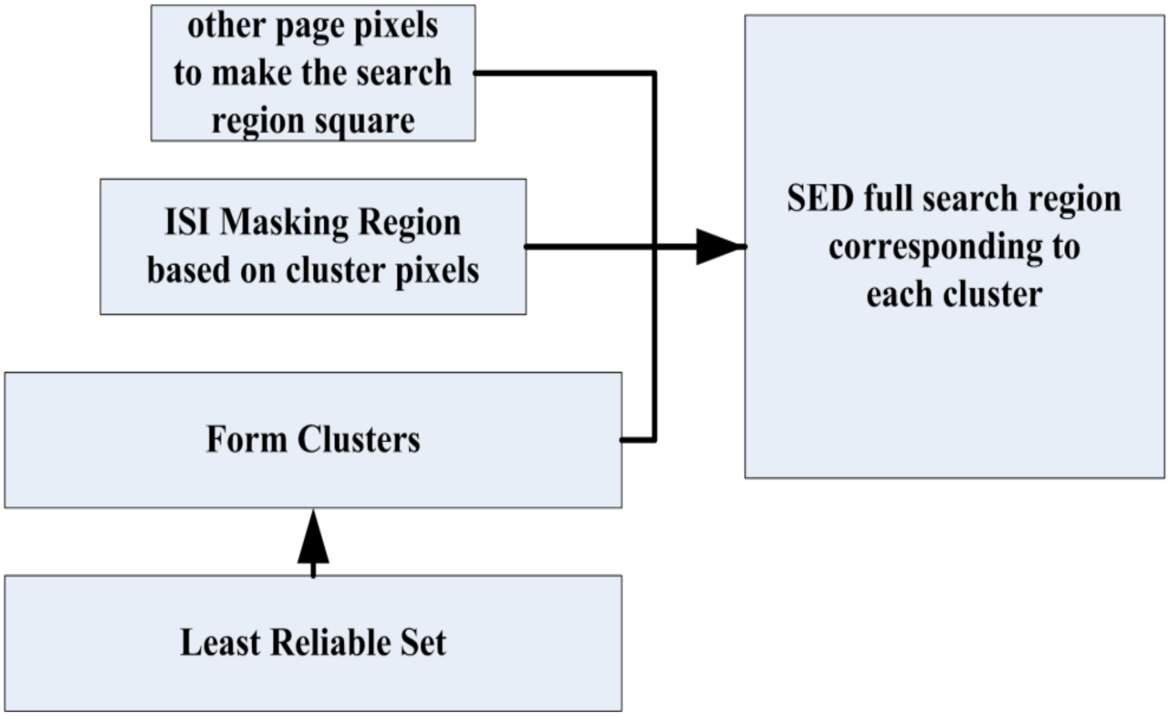
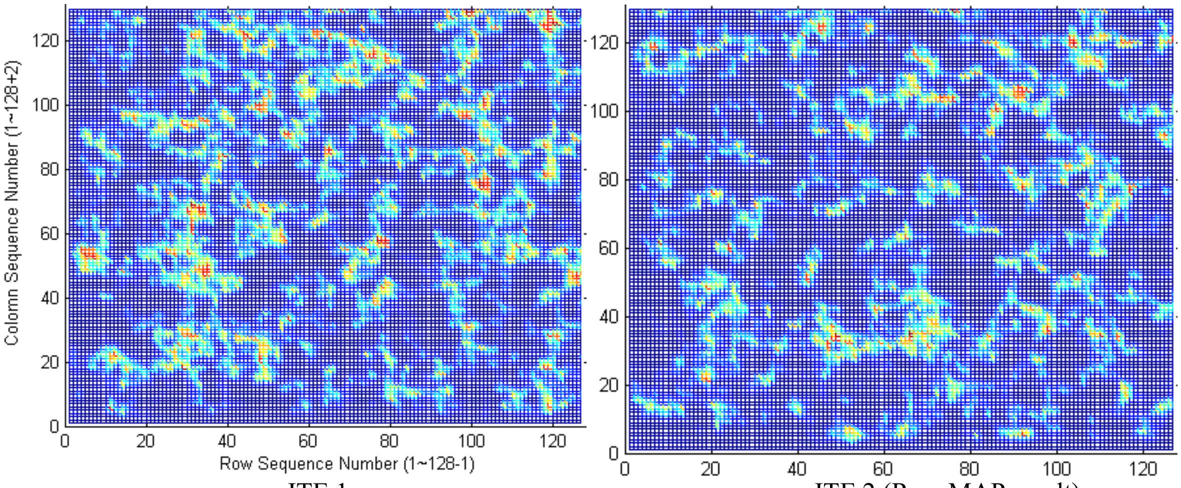

other page pixels to make the search region square
ISI Masking Region based on cluster pixels
Form Clusters
Least Reliable Set
SED full search region corresponding to each cluster

Fig. 24 Steps in identifying in SED based local search region

SED based local search for Basic Scheme: For a SED local search region, we assume that there is no TE in the ISI masking region; then for all possible combinations of cluster pixel values, we perform the convolution (2D inner product with the inverted mask), update the ISI result of the cluster pixels, and compute the SED of the updated SED local search region with the corresponding received set of pixel. The idea is that, the configuration of pixel values that, after convolution with mask, is closest to the received pixel values in terms of SED is (locally) optimal. Thus the final estimation result for the cluster pixels are set to be the configuration yielding the minimum SED value.

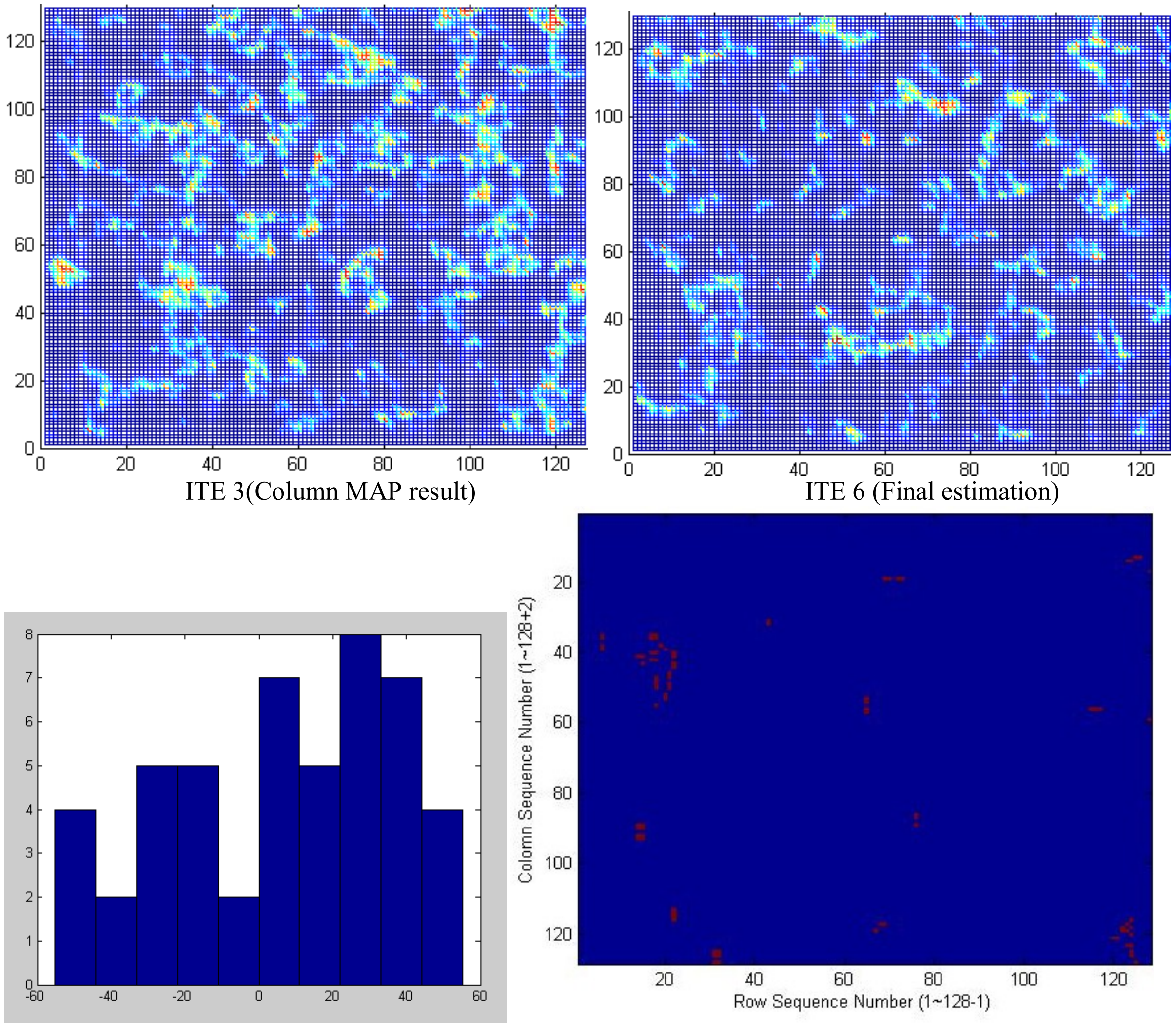
We use LLR thresholding to find the least reliable set. For basic scheme we used a LLR threshold T = T59. Then by an 8-connectivity check we eliminate all single pixels that are not adjacent to other set pixels, and thereby find all clusters. Like most 2D ISI equalization algorithms, the IRCSDFA has spatially correlated error patterns; this correlation means that most of the least-reliable pixels will be in clusters (i.e., not singletons), as shown in Fig.25.

After finding clusters, we assume that the boundary pixels around each cluster are correctly estimated. For a cluster of size L, we check all 2^L possible configurations of the cluster pixels, convolving the cluster configuration and its boundary pixels with the ISI mask and computing the SED with the corresponding local region of the received image **r**. The configuration with minimum SED is the final estimation for that cluster; this is likely to be the true value of the cluster, especially if the noise variance is small enough so that the clusters are isolated from one another and are not too large.



ITE 1

ITE 2 (Row MAP result)



ITE 3(Column MAP result)

ITE 6 (Final estimation)

Fig. 25 Illustration of LLR magnitude evolution along RC-BCJR iterations

In Fig.25, for all 89 TEs (the red pixels) in the final estimation page, although there is no strong relation between LLR magnitude evolution and the TE, the histogram shows that at the final stage of Row-ColumnBCJR iteration, LLR magnitude of the TEs are kind of evenly distributed on both low and high magnitude regions. But the spatial correlation is obvious.

The basic process is illustrated in Fig. 26. In Fig. 26 we partition the cluster pixels as: pixels in the set, cluster boundary pixels that are affected by convolution of the pixels in the set with the mask , and cluster boundary pixels that are not affected by ISI. It is possible to have true estimation errors (TEs) in both types of cluster boundaries. This basic process is performed on the final (sixth) iteration of the IRCSDFA, and the SED result is the final estimation result.

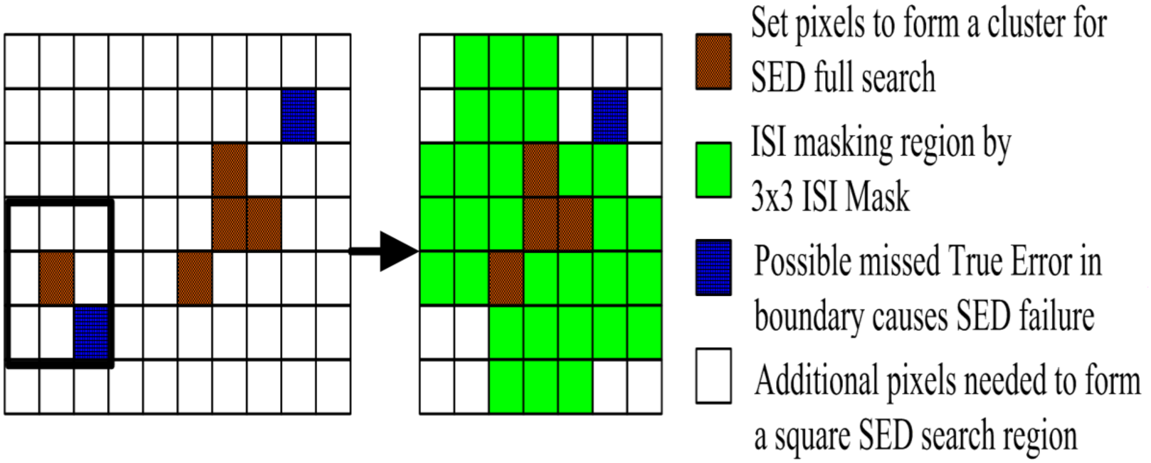

Set pixels to form a cluster for SED full search
ISI masking region by 3x3 ISI Mask
Possible missed True Error in boundary causes SED failure
Additional pixels needed to form a square SED search region

Fig. 26 Pixel and clustering in basic scheme

3.3 Motivation for the Advanced Scheme

As shown in Fig.26, in basic scheme we assume that the boundary has no TE, which is the major cause for errors not detected by SED search in basic scheme.

We observed that the noise variance is larger on clusters than over the whole page. At SNR = 15 dB, for the 3 × 3 averaging mask channel, the average noise standard deviation computed over pixels in 74 clusters from ten 128-by-128 input pages was equivalent to a SNR=12.74 dB. Due to the larger noise variance on clusters occasionally the SED search fails, but at this SNR experiments show that this kind of failure has a probability less than 0.1%. However, the SED search usually fails when there are one or more TEs in the cluster boundary, since the SED local search assumes that these boundary pixels have been correctly estimated by the IRCSDFA. Also note that the elimination of single pixels may also result in loss of some TEs. Based on these observations we propose an advanced process for better performance, with iterative cluster spanning and accumulative cluster overlapping.

3.4 The Advanced Scheme

Based on the basic scheme, several modifications are made to form the advance scheme:

1. Higher threshold *T*: For different *T*, e.g. T89 with T = 0.99999999 as threshold instead of T59, we get larger set size, more clusters and larger average cluster size. However, the performance gain is not much from T59 to T89, while the complexity exponentially increases. Thus in our experiments we stay with *T*=T59.

2. Iterative Cluster Span: To eliminate TEs in cluster boundaries and to save single TEs in the set that got singled out in the 8-connectivity check, we span the single pixels in the set as well as clusters iteratively. For iteration *N*, the N^{th} time we span a cluster, we compute

E = mean of all pixel LLRs in the cluster

STD = standard deviation of all of them,

and set threshold $T_{LLR} = E - N \times$ STD.

We then scan and include all boundary pixels with LLR less than T_{LLR} into the set, update the new cluster boundary and go to the *N*+1 iteration of cluster spanning.

This process can span single pixels in the set into a cluster, and can enlarge a cluster. The cluster iteration ends when no boundary pixels get included, due to fast decreases of T_{LLR}. The procedure of this iterative cluster span is illustrated in Fig. 29.

3. Treating Overlapping Clusters: As clusters grow larger, it is more and more possible for them to overlap with each other. To scan all clusters and check for overlapping is prohibitively complex, thus in our current version of the advanced process we use the estimation result from the previous clusters, whether overlapping or not.

All 3 factors in the advanced scheme is illustrated in Fig. 28. The advanced process is shown in Fig. 30 with a flow chart of the whole SED based local search post processing procedure in Fig. 27.

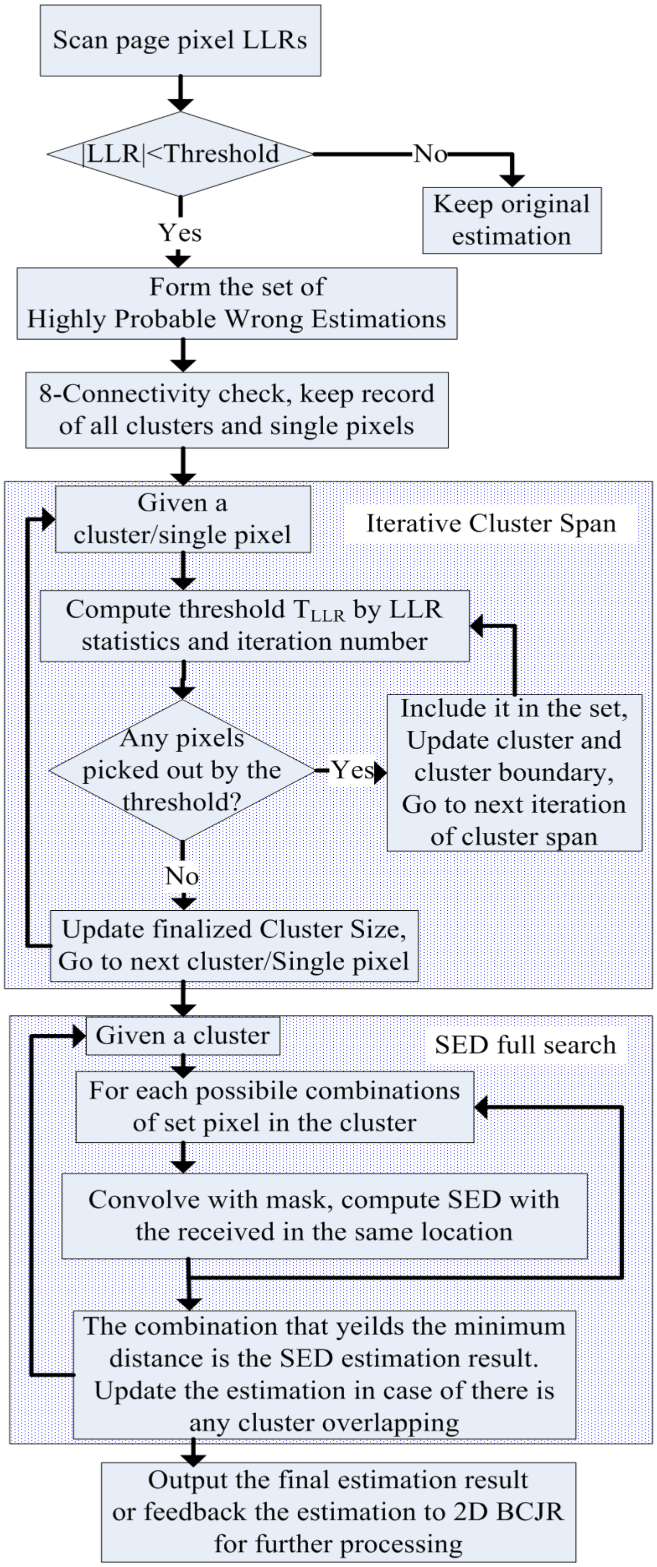
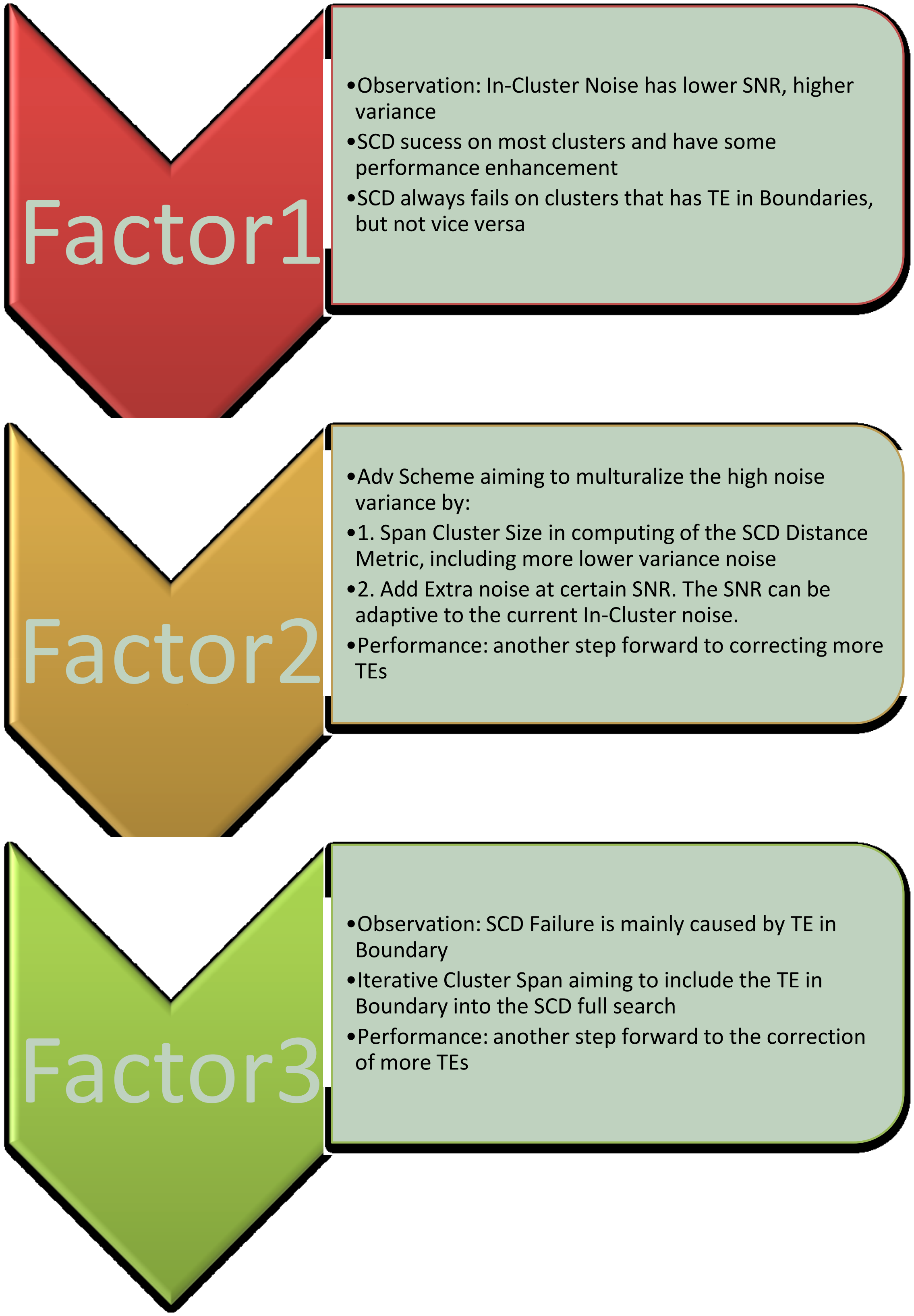

Scan page pixel LLRs
|LLR|<Threshold
No
Keep original estimation
Yes
Form the set of Highly Probable Wrong Estimations
8-Connectivity check, keep record of all clusters and single pixels
Given a cluster/single pixel
Iterative Cluster Span
Compute threshold T_{LLR} by LLR statistics and iteration number
Any pixels picked out by the threshold?
Yes
Include it in the set, Update cluster and cluster boundary, Go to next iteration of cluster span
No
Update finalized Cluster Size, Go to next cluster/Single pixel
Given a cluster
SED full search
For each possibile combinations of set pixel in the cluster
Convolve with mask, compute SED with the received in the same location
The combination that yeilds the minimum distance is the SED estimation result. Update the estimation in case of there is any cluster overlapping
Output the final estimation result or feedback the estimation to 2D BCJR for further processing

Fig. 27 Flow Chart for the SED based local search procedure



Factor1

- Observation: In-Cluster Noise has lower SNR, higher variance
- SCD sucess on most clusters and have some performance enhancement
- SCD always fails on clusters that has TE in Boundaries, but not vice versa

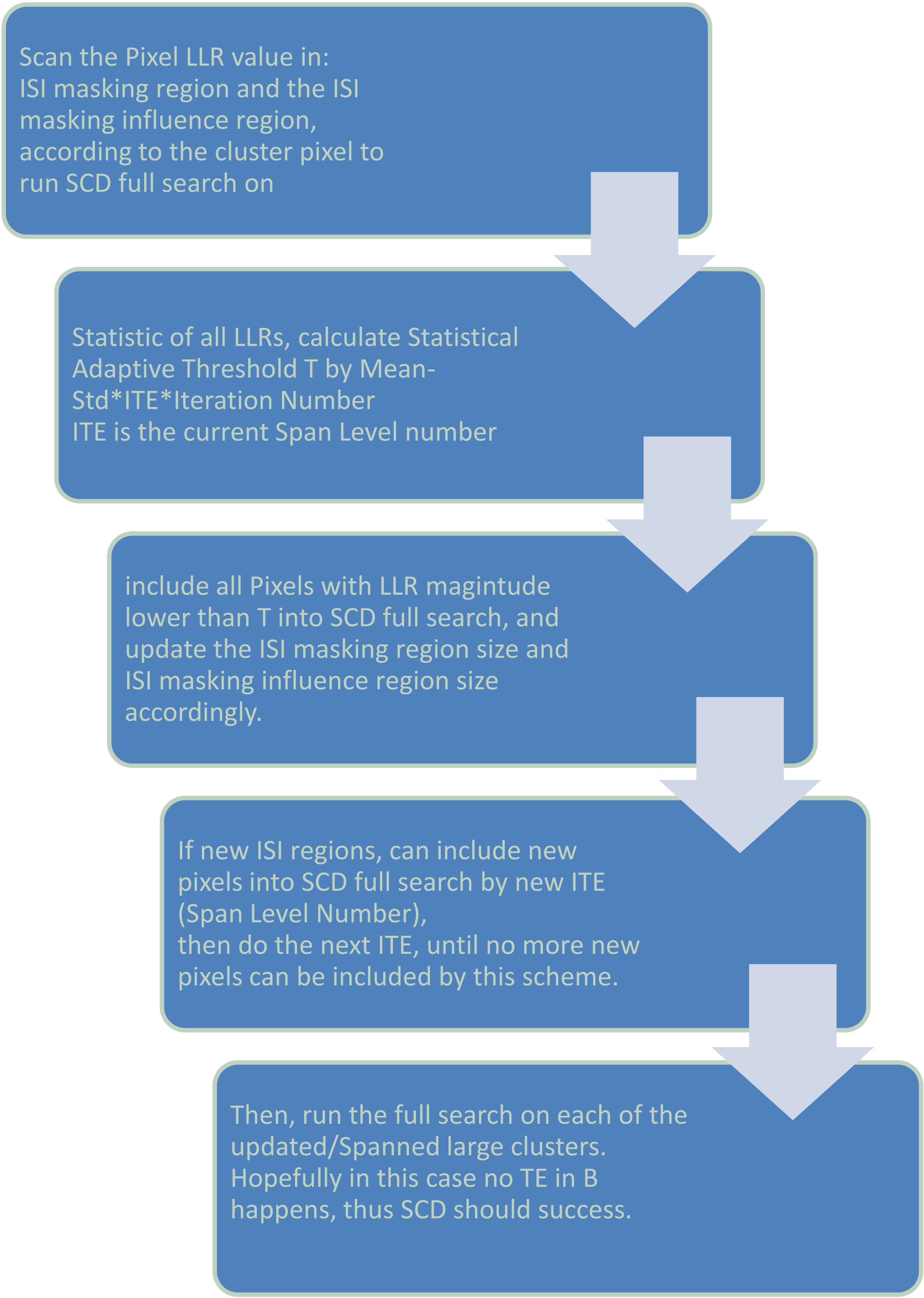
Factor2

- Adv Scheme aiming to multuralize the high noise variance by:
- 1. Span Cluster Size in computing of the SCD Distance Metric, including more lower variance noise
- 2. Add Extra noise at certain SNR. The SNR can be adaptive to the current In-Cluster noise.
- Performance: another step forward to correcting more TEs

Factor3

- Observation: SCD Failure is mainly caused by TE in Boundary
- Iterative Cluster Span aiming to include the TE in Boundary into the SCD full search
- Performance: another step forward to the correction of more TEs

Fig. 28 Conclusion of the 3 major factors in advanced scheme



Scan the Pixel LLR value in:
ISI masking region and the ISI masking influence region,
according to the cluster pixel to run SCD full search on

Statistic of all LLRs, calculate Statistical Adaptive Threshold T by Mean-Std*ITE*Iteration Number
ITE is the current Span Level number

include all Pixels with LLR magintude lower than T into SCD full search, and update the ISI masking region size and ISI masking influence region size accordingly.

If new ISI regions, can include new pixels into SCD full search by new ITE (Span Level Number),
then do the next ITE, until no more new pixels can be included by this scheme.

Then, run the full search on each of the updated/Spanned large clusters.
Hopefully in this case no TE in B happens, thus SCD should success.

Fig. 29 Conclusion of procedure in the iterative cluster span

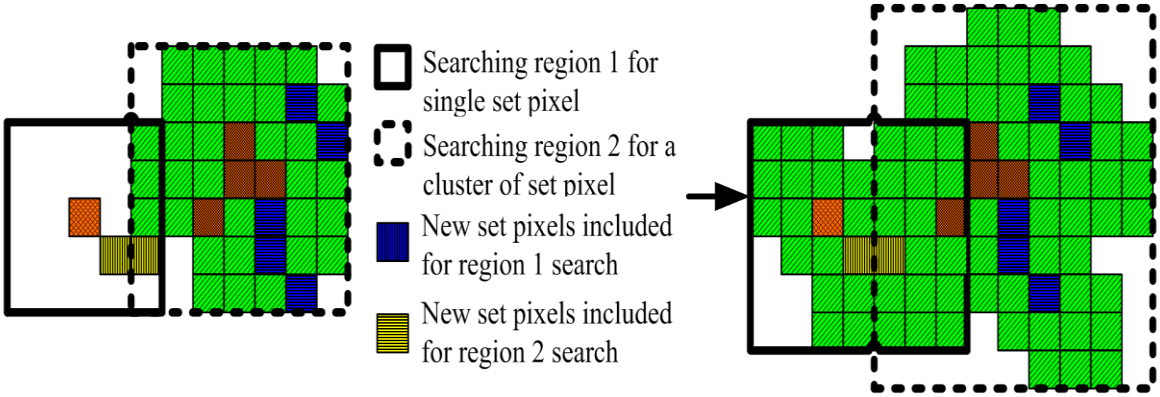

Searching region 1 for single set pixel
Searching region 2 for a cluster of set pixel
New set pixels included for region 1 search
New set pixels included for region 2 search

Fig. 30 Pixel and cluster processing in advanced scheme

3.5 Performance

In experiments at SNR=15 dB, the T59 threshold detected 33.68% of all TEs, with an average cluster size of 8.37 pixels. The basic process corrected 84.38% of the detected errors, but search failures introduced more errors in 12.16% of all clusters, with a final error correction rate of 18.95%.

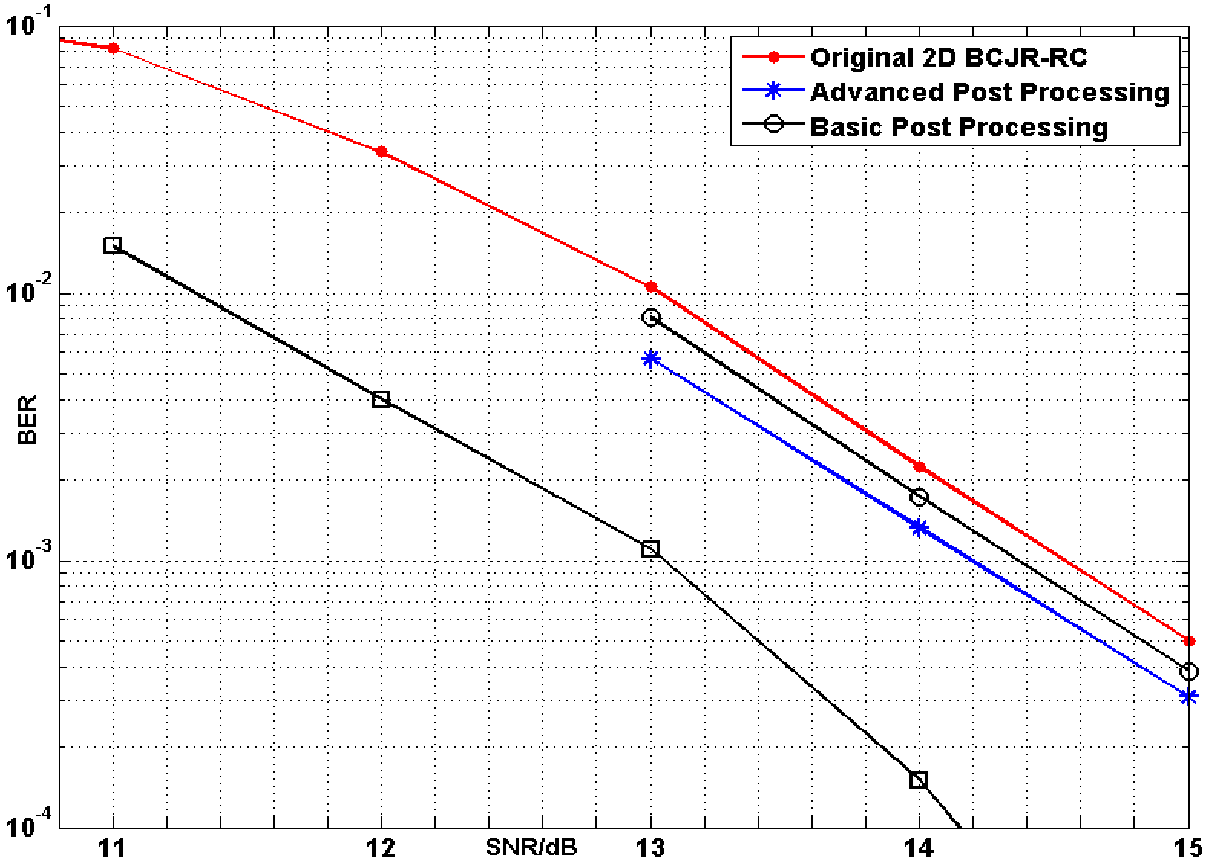


Fig. 31 Performance of the schemes of SED based local search algorithm

The advanced processing corrected all of the detected errors, but had search failures in 5.41% of all clusters, with a final correction rate of 28.43%, which is around a 0.4dB gain in performance. Fig. 31 compares the overall performance of the basic and advanced SED search schemes with that of the original 2D row-column BCJR acheme, for the 3 × 3 averaging mask channel.

The overall additional complexity for the SED local search was less than 12.5% of the basic row-column BCJR. On our computers, a single run of the original IRCSDFA at SNR=15 dB requires more than 40 minutes using C language code, while the SED search needed less than 5 minutes using Matlab code, running on the same machine.

3.6 Conclusion

In this chapter we introduced the SED based local search as post processing for IRCSDFA. Details about basic and advanced scheme for the SED search were presented. The proposed algorithm can enhance IRCSDFA performance by up to 0.4dB, with modest additional complexity required. Also note that the SED local search module can also be applied to other 2D ISI channel equalization schemes with error clustering characteristic for performance enhancement purpose.

CHAPTER FOUR: IMAGE DEBLURRING RESULT BASED ON IRCSDFA

In this chapter we apply the IRCSDFA with (*N*=2, *T*=0.99) as proposed in chapter one to the image deblurring. First we describe our image bluring and deblurring module, and then we present image result.

4.1 Image Blurring/Deblurring Model

For an input source color image, we can transform it into RGB or YCbCr color space, where each color channel is represented by 8-bit gray-scale pixels. Therefore without loss of generality, we focus on 8-bit gray-scale images.

For each of the eight bit-planes of the input source image, it is a binary input page with the same size. Each bit plane is convolved with the 3x3 averaging mask, followed by AGWN, and is then deblurred by IRCSDFA.

For bit plane of lower bit, the spatial correlation is weak and can be ignored, but for bit plane of higher bit, the spatial correlation is strong, but we still assume spatial independence and run the IRCSDFA on it. The decoding result shows that, there is seldom any decoding error on the higher bit plane, but there are errors on the lower bit plane. One explanation is that, higher bit plane has large clusters of pixels of the same value, thus is not interfered by the 2D ISI. The ISI interference only happen on intersections of clusters of pixel of same value, which takes a smaller fraction of all pixels, compared to the lower bit plane. Thus it is not likely for the higher bit plane to have decoding errors, and decoding errors on the lower bit plane does not affect the pixel value too much, thus will not deteriorate the performance by objective criteria (when viewed by real people).

The system structure is shown in Fig. 32.

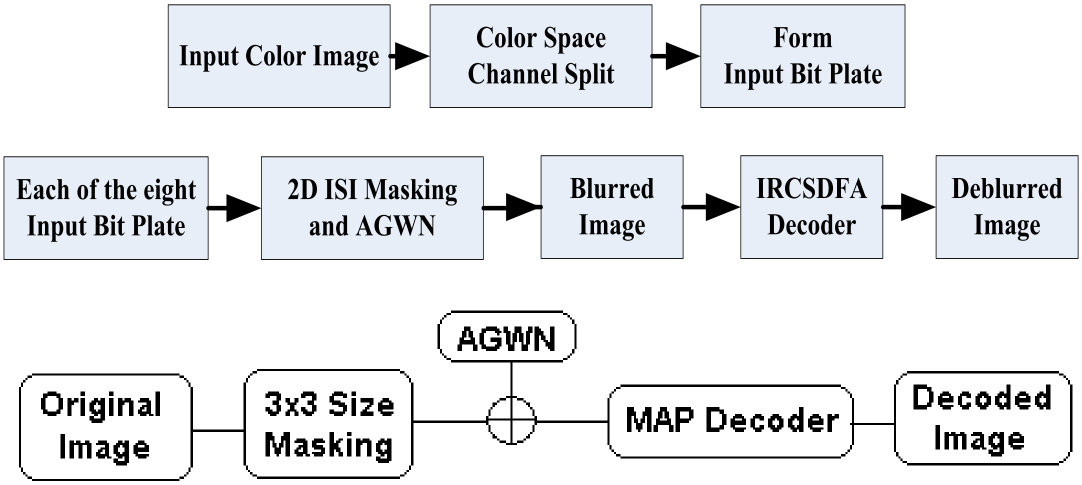


Fig. 32 System Structure for image blurring/deblurring by IRCSDFA

4.2 Simulation Image Result

We use the standard testing image Clown of 8-bit gray image with size 128-by-128 in Fig.33 to show the image result.

The SNR only influence the amount of noise in the AGWN generator. Fig.34 shows that at high SNR =15 dB, the blur is small, and IRCSDFA gives only 47 pixel errors in the decoded image in Fig.33.

Fig.36 shows that at low SNR = 10 dB the blur is relatively large, and IRCSDFA gives 277 pixel errors in decoded image in Fig.37 (row detection result). It is easy to see that due to the spatial correlation of errors, the error pixels usually form a 2-by-2 cluster.

Fig.38 shows a better image (post processed detection result) result by objective criteria, which is achieved by 3x3 averaging on all least reliable clusters (by threshold T59).

The IRCSDFA used here is the complexity reduced algorithm with parameter (N=2, *T*=0.99) with equivalent performance as shown in chapter one. With over 96% of the complexity reduced, the decoding time is also greatly reduced, and therefore the proposed system is capable of processing large color images. Note that based on our way to split color space channels and split bit planes, for a deblurring of a color image in RGB or YCbCr format with 8-bit precision, it requires 24 such systems as in Fig. 30 working in parallel to get a decoding result. And if the input image is of 16-bit precision as in medical image format, we will be needing 48 such systems in parallel to deblur. This parallel structure and the greatly reduced amount of computation make this image deblurring system a promising application future in the multi-core multi-thread or blade computing system.

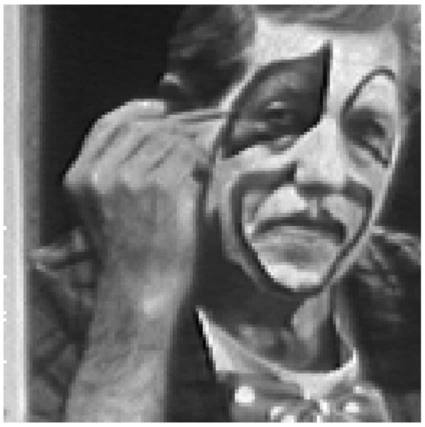


Fig. 33 Original Testing Image “Clown”

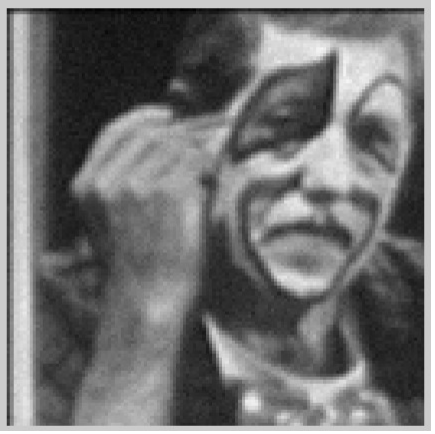


Fig. 34 Corrupted Image with noise at

SNR=15 dB

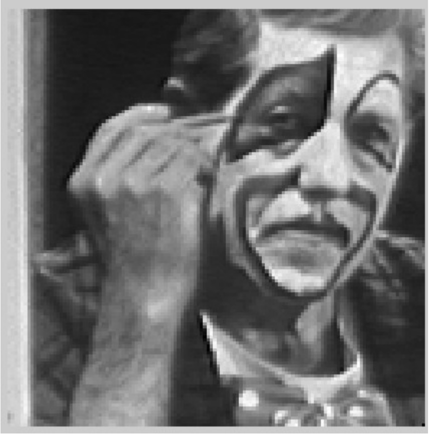


Fig. 35 Detection Result at SNR=15 dB,

with only 47 none zero pixel error

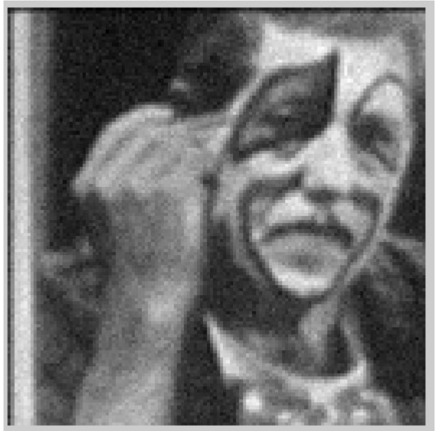


Fig. 36 Corrupted Image with noise at

SNR=10 dB

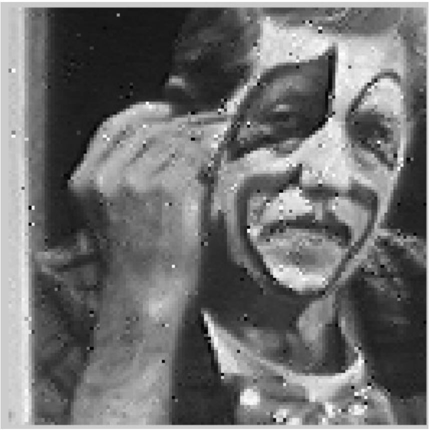


Fig. 37 Raw Detection Result without post processing a SNR=10 dB

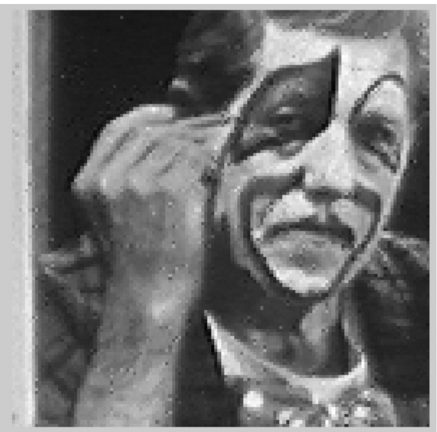


Fig. 38 Post Processed Detection Result at SNR=10 dB

CHAPTER FIVE: CONCLUSION AND FUTHURE WORK

In this thesis we present two improvements to the IRCSDFA of [14].

For complexity reduction we proposed a feedback probability based sorting algorithm, with over 96% effective complexity reduction and almost no performance deterioration for the basic scheme, and for the advanced scheme we have up to 0.25dB of performance gain with 1.2 times of complexity of the original algorithm, and up to 0.19dB of performance gain and over 80% of complexity reduction.

For performance enhancement we proposed a SED based local search post processing on the output of the IRCSDFA, with up to 0.4dB performance gain and the additional complexity is less than 12.5% of the IRCSDFA's complexity.

For further work on the application part, the proposed image deblurring scheme with parallel structure, which is frasible in practice, thanks to the computational reduction achieved by the FbPr sorting algorithm, is an interesting direction with encouraging preliminary deblurring results.

For further work on the algorithm part, the parameter set of both algorithms needs to be optimized for random inputs. Currently we are experimenting with genetic algorithms (GAs) to find optimized parameter sets with the best detection performance. The GA requires 300~500 single runs to converge, and so is quite time consuming to run, especially at high SNR. The best parameters found by our GA trained on ten random input pages are shown in Table 1. These parameters are very close to the parameters we used in our SED search experiments, and both give very similar performance, as in Fig.39. The one main difference by far is that the iterative cluster span LLR threshold function found by GA is not similar to the one used in the algorithm proposed in chapter three. One explanation might be that, the results shown by GA is based only on 10 random input pages, while even on high SNR =15 dB in order to get 100 error counts we always need to simulate more than 15 input pages.

We believe further parameter optimization will lead to improved performance of the algorithm.

Item	Application	Optimized parameter
LLR Threshold to form the set	Globally optimized	T_{OPT}=0.9999998
	Actually Used	T=T59 i.e. T=0.99999
LLR Threshold in Iterative cluster span	Globally optimized	$T_{\mathrm{LLR\text{-}OPT}}$= 1.0318*E + (0.8997 * N + 0.0042 * N^2) * STD
	Actually Used	T_{LLR}= E + N*STD

Fig. 39 Preliminary Results on Genetic Algorithm Optimization

BIBLIOGRAPHY

[1] H. Ma, K. Sivakumar, and B. J. Belzer, "Feedback probability sorting and LLR-based SED search for iterative row-column 2D ISI equalization," in *Proc. 44rd annual Conference on Information Sciences and Systems (CISS 2010)*, Princeton, NJ, 2010.

[2] W. Coene. (2004, Mar.) Coding and signal processing for two-dimensional optical storage (TwoDos). [Online]. http://cm.bell-labs.com/cm/ms/events/WGIR04/pres/coene.ppt

[3] G.T. Huang, "Holographic memory," *MIT Technology Review*, vol. 9, Sept. 2005.

[4] C. L. Miller, B. R. Hunt, M. W. Marcellin, and M. A. Neifeld, "Image restoration using the Viterbi algorithm," *J. Opt. Soc. Amer. A*, vol. 16, no. 2, pp. 265-274, Feb. 2000.

[5] G. D. Forney, "The Viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268-278, Mar. 1973.

[6] E. Ordentlich and R. M. Roth, "On the computational complexity of 2D maximum-likelihood sequence detection," Hewlett-Packard Technical Report HPL-2006-69, 2006.

[7] K. M. Chugg, "Performance of optimal digital page detection in a two-dimensional ISI/AWGN channel," in *Proc. Asilomar Conf. Signals, Systems and Computing*, 1996, pp. 958-962.

[8] M. Marrow, P. H. Siegel, and J. K. Wolf J. B. Soriaga, "On achievable rates of multistage decoding on two-dimensional ISI channels," *Proc. 2005 IEEE Int. Symp. on Info. Theory*, p. 1348–1352, Sept. 2005.

[9] J. Chen and P. H. Siegel, "On the symmetric information rate of two- dimensional finite-state ISI channels," *IEEE Trans. Inf. Theory*, vol. 52, no. 1, p. 227–236, Jan. 2006.

[10] K. Gürkan, and L. Hesselink J. F. Heanue, "Signal detection for page- access optical memories with intersymbol interference," *Applied Optics*, vol. 35, no. 14, p. 2431–2438, May 1996.

[11] R. Krishnamoorthi, "Two-dimensional Viterbi like algorithms," *Master's thesis, Univ. Illinois at Urbana Champaign*, 1998.

[12] J. A. O'Sullivan, N. Singla, and Ronald S. Indeck Y. Wu, ""Iterative detection and decoding for separable two-dimensional intersymbol interference"," *IEEE Trans.Magnetics*, vol. vol.39 no.4, pp. 2115-2120, July 2003.

[13] M. Marrow and J. K. Wolf, "Iterative detection of 2-dimensional ISI channels," in *Proc. Inform. Theory Workshop*, Paris, France, 2003, pp. 131-134.

[14] T. Cheng, B. J. Belzer, and K. Sivakumar, "Row-column soft-decision feedback algorithm for two-dimensional intersymbol interference," *IEEE Sig. Proc. Letters*, vol. 14, pp. 433-436, July 2007.

[15] Patrick Njeim, Taikun Cheng, Ben Belzer and K. Sivakumar Yiming Chen, "Iterative Soft Decision Feedback Zig-Zag Equalizer for 2D Intersymbol Interference Channels," *IEEE Journal on selected areas in comm*, vol. 28, no. 2, Feb. 2010.

[16] 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, no. 2, pp. 284-287, Mar. 1974.

[17] Y. Zhu, B. J. Belzer, and K. Sivakumar, "Reduced complexity iterative equalizers for M-ary 2D ISI channels," in *Proc. 43rd Conf. Info. Sci. and Syst. (CISS 2009)*, Baltimore, MD, 2009, pp. #216 (CD-ROM).

[18] K. Sivakumar, and Benjamin Belzer Hannan Ma, "Feedback Probability Sorting and LLR-based SED Search for Iterative Row-Column 2D ISI Equalization," in *Proc. 44th Conf. Info. Sci. and Syst. (CISS 2010)*, Princeton, NJ, 2010.