Huawei Report: A Survey of Turbo MIMO Receivers for LTE/LTE-A

Students: Peter Kairouz and Eric Kim

Faculty: Naresh Shanbhag and Andrew Singer

Coordinated Science Laboratory / Department of Electrical and Computer Engineering

University of Illinois at Urbana Champaign, Urbana, Illinois, 61801

Email: {kairouz2, epkim2, shanbhag, acsinger}@illinois.edu

Abstract

Since their emergence, turbo multiple-input multiple-output (MIMO) systems experienced many advancements on both the transmitter and receiver sides. In this report, we review the major contributions in the field of iterative MIMO receivers. Sections II, III, IV, and V provide an extensive review on current hard and soft detectors, channel decoders, and iterative receivers for MIMO systems. The presented algorithms are compared from a computational complexity versus performance perspective. In section VI, we propose a set of research directions for the Huawei/UIUC turbo MIMO receivers project. These research directions include turbo scheduling, receiver algorithm development, and implementations of turbo MIMO receivers.

I. MIMO System Model

A MIMO system has M_T transmit antennas that transmit a $M_T \times 1$ symbol vector $\mathbf{s}$ each channel use. Each element of $\mathbf{s}$ is transmitted on one antenna and is chosen from a constellation of size 2^q. Thus, a total of qM_T bits are transmitted per channel use. The set of constellation points are denoted as $\mathcal{F}$ and the transmit vector $\mathbf{s} \in \mathcal{F}^{M_T}$. The channel is modeled as a flat Rayleigh fading channel such that the received vector is $\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n}$. Denoting M_R as the number of received antennas, $\mathbf{y}$ is a $M_R \times 1$ vector, $\mathbf{H}$ is the channel matrix of size $M_R \times M_T$ with channel coefficients typically modeled as independent and identically distributed (i.i.d.) complex Gaussian random variables, and $\mathbf{n}$ is spatially white Gaussian noise, a $M_R \times 1$ complex Gaussian vector with zero mean and variance chosen for a specific SNR. This model is shown in Fig. 1. Note: the i.i.d. assumption on $\mathbf{H}$ is a baseline assumption. In practice, we will employ both i.i.d. and correlated channel models in our study. Nevertheless, it should be noted that the MIMO capacity will be reduced in the presence of correlated channel entries. Finally, MIMO gains could be enhanced via several techniques such as spatial multiplexing and transmit diversity, in addition to others. Actually, Long Term Evolution Advanced (LTE-A) systems, the focus of this project, provides the designer the choice of using either transmit diversity or spatial multiplexing (or both). Hence, we will consider both modes of operation. When appropriate, we will describe how the transmitted vector $\mathbf{x}$ is chosen to satisfy one or both modes of operation.

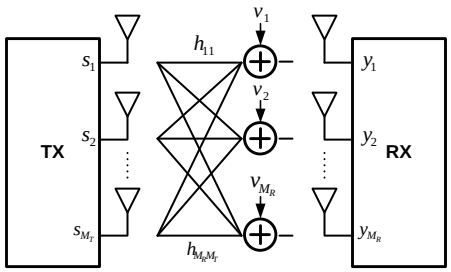


Fig. 1. Block diagram of a MIMO wireless link.

II. Hard MIMO Detectors

A. Introduction

The goal of the MIMO detector is to separate out the original transmitted signal from the received signal. The detection process can be separated into two stages, a preprocessing stage where the channel matrix $\mathbf{H}$ is calculated and other channel related processing is implemented, and a detection stage. Typically the preprocessing stage is implemented at the channel variation rate, while the detection stage needs to operate at the transmit data rate (See Fig. 2).

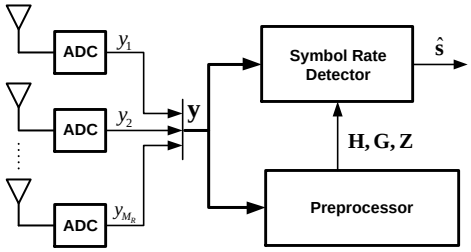


Fig. 2. Channel rate preprocessing and symbol rate detection of a MIMO detection system.

One important role of the preprocessor is to perform channel estimation. Channel estimation provides the receiver with channel state information (CSI), i.e., channel matrix $\mathbf{H}$. The optimal detector in terms of bit error rate (BER) is the maximum likelihood (ML) detector, which is given by

$$\hat{\mathbf{s}}_{ML} = \underset{\mathbf{s}\in\mathcal{F}^{M_T}}{\arg\min} \|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 \tag{1}$$

The ML detector finds the nearest neighbor among all possible constellation points to the received vector.

Given that the ML detector is the optimal MIMO detector, the goal of all MIMO detectors is to approach ML performance with minimal computational complexity. The ML detector is an exhaustive search algorithm which is exponential in the constellation size $\mathcal{F}$ and the number of antennas M_T. Based on complexity, there are four well known classes of detectors:

1) Linear detectors : zero forcing (ZF) and minimum mean squared error (MMSE)
2) Nonlinear, non ML based, sub-optimal detectors : decision feedback detection and (ordered) successive interference cancelation (OSIC)
3) Nonlinear, ML based, optimal detectors : tree-based algorithms such as sphere decoding
4) Nonlinear, ML based, sub-optimal detectors: K-best detection, reduced dimension ML (RDML) detection, and combined ML and DFE detection

B. Linear Detectors

In spatial multiplexing, the MIMO system can be thought of as a system of linear equations. There are M_T unknowns with M_R equations. For $M_R > M_T$ it is an over-determined system, and the minimum squared error solution is desired. ZF and MMSE perform detection as if solving a system of linear equations. Thus, a linear MIMO receiver is described by

$$\hat{\mathbf{x}} = \mathbf{G}\mathbf{y} \tag{2}$$

$$\hat{\mathbf{s}} = Q[\hat{\mathbf{x}}] \tag{3}$$

$$\tag{4}$$

where $Q[\cdot]$ is a slicer function. Depending on the choice of $\mathbf{G}$, several linear algorithms exist.

1) Zero Forcing (ZF): In zero forcing, $\mathbf{G}$ is chosen to be the *Moore-Penrose pseudo-inverse* or the pseudo-inverse of the channel matrix (the inverse of $\mathbf{H}$ will not exist in general as it may not be a square matrix).

$$\hat{\mathbf{x}} = \mathbf{H}^{\dagger}\mathbf{y} = \mathbf{s} + \mathbf{H}^{\dagger}\mathbf{v} \tag{5a}$$

$$\hat{\mathbf{s}} = Q[\hat{\mathbf{x}}] \tag{5b}$$

$$\mathbf{H}^{\dagger} = (\mathbf{H}^{*}\mathbf{H})^{-1}\mathbf{H}^{*} \tag{5c}$$

$()^{*}$ denotes the conjugate transpose operation.

2) Minimum mean squared error (MMSE): The MMSE MIMO receiver is described as

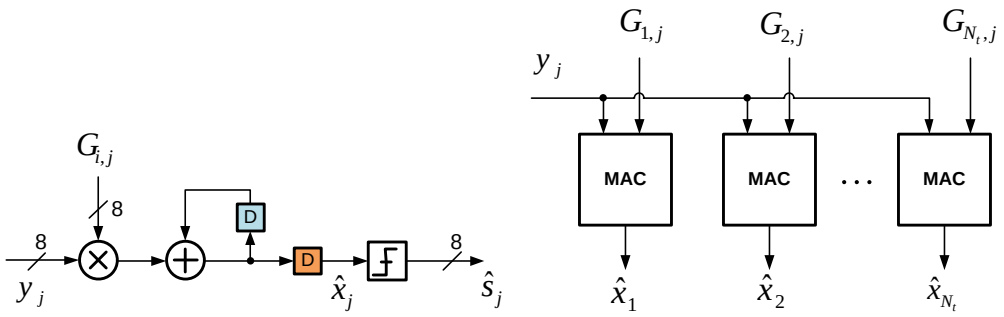
$$\hat{\mathbf{x}} = \mathbf{G}\mathbf{y} = \mathbf{G}\mathbf{s} + \mathbf{G}\mathbf{v} \tag{6a}$$

$$\hat{\mathbf{s}} = Q[\hat{\mathbf{x}}] \tag{6b}$$

$$\mathbf{G} = (\mathbf{H}^{*}\mathbf{H} + \sigma_v^2 M_T \mathbf{I}_{M_T \times M_T})^{-1}\mathbf{H}^{*} \tag{6c}$$

where σ_v^2 is the noise variance, and $\mathbf{G}$ can be viewed as a noise-biased pseudo-inverse, and the MMSE algorithm as a biased estimator.

Both ZF and MMSE perform matrix multiplication in the detection stage. A matrix multiplier is conventionally implemented using a multiply and accumulate (MAC) unit. The block diagram of a MAC unit, and the implementation of a ZF detection stage is depicted in Fig. 3. Parallel MAC units can be employed to speed up the computation.



(a) (b)

Fig. 3. Architecture of: (a) multiply-accumulate (MAC) unit, and (b) linear detection stage.

C. Successive Interference Cancellation

Successive interference cancellation (SIC) relies on estimating one of M_T transmitted symbols first, say s_1 and then subtracting $\hat{s}_1$ from the received stream. This process is repeated M_T times in order to detect the transmitted vector $\mathbf{s}$. The SIC algorithm is described by Algorithm 1.

Ordered SIC is a variant of SIC in which the order in which the transmitted symbols are detected is specifically chosen to minimize the BER. For example, an observation of the channel matrix $\mathbf{H}$ might reveal that the norm of the j^{th} column $\|h_j\|$ is the maximum over all M_T columns. In this case, s_j has the highest symbol energy across the M_T received antennas, and can be detected with the highest reliability. Thus, ordered SIC sequences the detection from the most reliable transmitted symbol first to the least reliable last.

For SIC, a block similar to the matrix multiplication used in ZF can be used for each iteration, but the iterations cannot be done in parallel. Thus several chains of iterations can be scheduled using pipelining to match the throughput of a parallel implementation of ZF. The architectural block diagram for a single iteration, and for the complete detection stage is shown in Fig. 4.

1) ZF-DFE: The ZF-DFE or ZF-SIC detection scheme is a variant of SIC. Actually, this technique was suggested in [1] for the Vertical Bell Laboratories Layered Space-Time (V-BLAST) system. This method has three main features:

- interference cancellation (feedforward filtering) to cancel the interference of different subchannels
- interference nulling (feedback filtering) to null out the interference of different subchannels

Input: received vector $\mathbf{y}$, channel matrix $\mathbf{H}$

Output: transmitted symbol $\hat{\mathbf{s}}$

begin

$i \longleftarrow 1$

$\mathbf{y}^{(i)} \longleftarrow \mathbf{y}$

$\mathbf{H}^{(i)} \longleftarrow \mathbf{H} = [\mathbf{h}_i, \mathbf{h}_{i+1}, \ldots, \mathbf{h}_{M_T}]$

for $i \leftarrow 1$ **to** M_T **do**

$\mathbf{G}^{(i)} \longleftarrow (\mathbf{H}^{(i)*}\mathbf{H}^{(i)} + \sigma_v^2 \mathbf{I}_{(M_T-i+1)\times(M_T-i+1)})^{-1}\mathbf{H}^{(i)*}$

$\hat{x}_i \longleftarrow \mathbf{g}_i^{(i)}\mathbf{y}^{(i)}$

$\hat{s}_i \longleftarrow Q[\hat{x}_i]$

$\mathbf{y}^{(i+1)} \longleftarrow \mathbf{y}^{(i)} - \hat{s}_i \mathbf{h}_i$

end

end

Algorithm 1: Algorithm for successive interference cancelation (SIC). $\mathbf{g}_i^{(i)}$ is an $1 \times M_R$ vector representing the i^{th} row of the $(M_T - i + 1) \times M_R$ matrix $\mathbf{G}^{(i)}$.

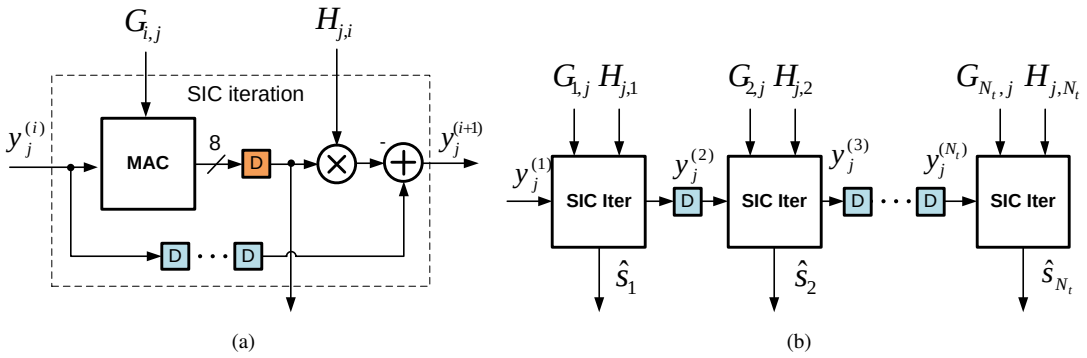


Fig. 4. Architecture diagram of: (a) a single iteration stage, and (b) complete detection stage for a successive iteration cancelation receiver.

- subchannel ordering to reduce the effect of error propagation

Assuming that the received signal is of the form $\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{v}$, and that the matrix $\mathbf{H}$ is known at the receiver, we can always decompose $\mathbf{H}$ into the product of a unitary matrix ($\mathbf{Q}$) and an upper triangular matrix ($\mathbf{R}$) via the QR factorization. Thus, for the interference cancellation stage we left multiply the received vector $\mathbf{y}$ by $\mathbf{Q}^*$ to obtain $\tilde{\mathbf{y}} = \mathbf{Q}^*\mathbf{y} = \mathbf{Q}^*(\mathbf{Q}\mathbf{R}\mathbf{s} + \mathbf{v}) = \mathbf{R}\mathbf{s} + \tilde{\mathbf{v}}$. After this step it easy to see that the decoder can begin detecting symbols sequentially after detecting the symbol s_{M_T} from $y_{\tilde{M}_T} = r_{M_T,M_T} s_{M_T} + \tilde{v}_{M_T}$ and nulling out its effect form the others (interference nulling).

SIC, although simple and effective, suffers from what is known as error propagation since an error in decoding any symbol will impact the detection of subsequent symbols. In fact, it can be shown that the performance of this system is limited by the detection of the n^{th} symbol (the n^{th} subchannel) since the statistics of $r_{n,n}$ are the worst ($r_{n,n}$ is χ square distributed with the least degrees of freedom). This is why improving the decoding of the n^{th} subchannel is crucial to improving the system performance.

D. Maximum Likelihood

The optimal MIMO detector is the maximum-likelihood (ML) detector. ML detector finds the most probable symbol $\hat{\mathbf{s}}$ that could have been transmitted given the received vector $\mathbf{y}$. Under the assumption of additive white Gaussian noise, and equally probable prior distribution for the symbol $\mathbf{s}$, the ML detector chooses the nearest neighbor. Thus the ML detector finds the solution to $\arg\min_{\mathbf{s}\in\mathcal{F}^{M_T}} \|\mathbf{y} - \mathbf{H}\mathbf{s}\|$. A straightforward implementation will perform an exhaustive search over all $\mathcal{F}^{M_T}$ possible transmit vectors, which is prohibitively costly for high dimension systems. ML achieves the optimal BER performance at the cost of exponential complexity.

E. Sphere Decoding

ML detection is an NP-hard problem, which means that all known solutions have a worst-case complexity that is exponential in M_T. However, the average complexity of a MIMO detector depends on the SNR, M_T, and M_R. Because the received vector $\mathbf{y}$ is not arbitrary, but a point obtained by perturbing $\mathbf{G}\mathbf{x}$ with Gaussian noise $\mathbf{n}$, the average complexity is much less than the worst-case exponential complexity.

Many sophisticated algorithms have been developed to solve the ML problem. Historically the most widely used algorithm is the Viterbi algorithm. However this algorithm is limited to Toeplitz matrices and is often used for ISI SISO channels. There are other algorithms such as Kannan's algorithm (which performs a search within a parallelogram) [2], KZ algorithm [3] (which utilizes the Korkin-Zolotarev reduced basis), and the Fincke and Pohst sphere decoding (SD) algorithm [4]. SD is the most efficient and attains polynomial time complexity in a large number of cases (though the worst case is still exponential).

The basic idea for SD is to perform search over lattice points that lie within a sphere centered at the received vector $\mathbf{y}$. It is obvious that the closest lattice point within the sphere to the received vector r is also the closest point among all the lattice points. This gives rise to two questions:

1) What should the radius r be? Determining the proper radius is important because, if the radius is too large, the number of points needed to search for will be large, while if the radius is too small, no lattice points may exist within the sphere.

2) Which points of the space lie within the sphere? This can be answered by testing the distance of each lattice point to the received vector. Without a simple method to determine which points lie in the sphere, SD will be no different from the original ML algorithm as every lattice point will need to be tested to determine if they lie within the sphere or not.

SD does not give the answer to the first question, but it provides an efficient algorithm to determine the answer to the second question. The basis of the algorithm is to transform the lattice points into a tree. A 16-QAM modulation (with constellation points $\pm 3 \pm 3j, \pm 3 \pm j, \pm 1 \pm j, \pm 1 \pm 3j$) with a 2-dimensional ($M_T = 2$) sphere can be represented as shown in Fig. 5. As seen in the figure, each branch corresponds to a constellation point, while the depth of the tree equals the dimension. The number of leaves equal $|\mathcal{F}^{M_T}|$, which is 16^2 for this example. By traversing the tree from the root to the leaf, we obtain the complete symbol vector of a specific lattice point.

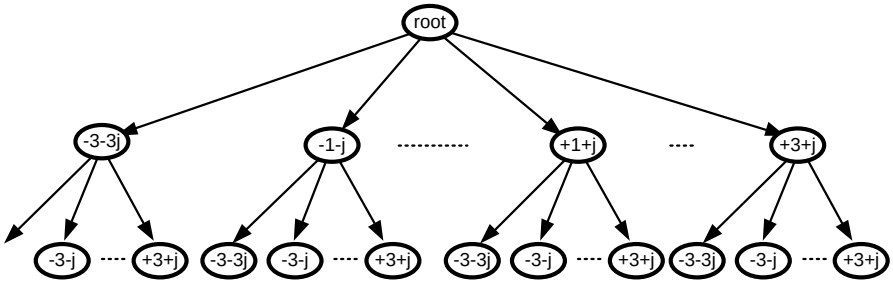


Fig. 5. Tree representation of a 16-QAM, 2-dimensional lattice.

Now if the distance from the received vector to a specific lattice point (the leaf of a tree) can be obtained by adding the distance of each branch, we have an efficient tree based algorithm of determining which points lie within the sphere. If the distance from the root to a particular node in the tree exceeds the sphere radius, we can conclude that all leafs that have this particular node as a parent will lie outside the sphere. Thus we can successfully prune all the branches below this node.

The tree representation of the lattice points does indeed have this branch metric property and the distance from the root to a particular node is called the partial Euclidean distance (PED). While traversing down the tree, PED is a non-decreasing value and the PED at the leaf equals the distance between the received vector and the lattice point. To obtain the PED, the ML rule needs to be transformed. This process only needs to be done once for a specific channel $\mathbf{H}$ and thus is often done in the preprocessing stage.

1) Preprocessor: Tree based detection algorithms first transforms the ML decoding rule through the QR decomposition of the channel $\mathbf{H} = \mathbf{QR}$, where $\mathbf{Q}$ is unitary and $\mathbf{R}$ is upper triangle. The QR decomposition is done as

part of the preprocessing stage. As the norm is invariant to unitary transforms, the ML rule can be transformed as follows

$$\hat{\mathbf{s}} = \underset{\mathbf{s}\in\mathcal{F}^{M_T}}{\arg\min}\|\mathbf{y}-\mathbf{H}\mathbf{s}\| = \underset{\mathbf{s}\in\mathcal{F}^{M_T}}{\arg\min}\|\mathbf{Q}^H\mathbf{y}-\mathbf{R}\mathbf{s}\| = \underset{\mathbf{s}\in\mathcal{F}^{M_T}}{\arg\min}\|\tilde{\mathbf{y}}-\mathbf{R}\mathbf{s}\| \tag{7}$$

where $\tilde{\mathbf{y}} = \mathbf{Q}^H\mathbf{y}$. Due to the triangular structure of $\mathbf{R}$, the norm can now be rewritten as a sum of several vector norms

$$\|\tilde{\mathbf{y}}-\mathbf{R}\mathbf{s}\| = \sum_{i=1}^{M_T}\left|\tilde{y}_i - \sum_{l=i}^{M_T} r_{i,l}s_l\right|^2 \tag{8}$$

As the last $M_T - i + 1$ elements of summation only depend on the last $M_T - i + 1$ transmitted symbols and the summation is non-decreasing, if this partial sum (partial Euclidean distance) exceeds the specified radius of the sphere, it is safe to drop all candidate vectors that end with the same sequence of symbols. This is referred to tree pruning.

2) Sphere Detector: Recently, tree-based sphere detectors (SD) have gained much interest as it enables the search space to be restricted to the points that lie within a certain hyper-sphere centered around $\mathbf{y}$. SD reduces the complexity of ML significantly, but still the expected number of elements in the search space is still exponential in the dimension of the MIMO system, thus application of SD at high dimensional systems are of great concern. Sphere decoding (SD) algorithm has a close relation with tree based search algorithms where each node is associated with a constellation point and traversing the tree from root to leaf gives the complete transmit vector. When a sphere condition is violated, it can be thought of as pruning all branches below the current node. Through this process, SD is capable of significantly reducing the search set.

3) Radius: The sphere radius is an important parameter that has considerable effect on the runtime complexity of SD. For the SD to produce a meaningful result, the radius must be large enough to contain at least one leaf node. However, if the radius is too large, SD converges to an exhaustive search ML. One reasonable value for the radius is to use the distance between y and the ZF or MMSE solution. However, for ill conditioned channels, many points may lie near the ZF solution and still cause the SD to iterate over a significant number of nodes. To even further speed up the SD process, a dynamic radius maybe used where the radius is constantly updated to the minimum value. This enables a larger number of nodes to be pruned resulting in lower complexity/higher throughput. In dynamic radius reduction, the initial radius is often set to the distance of the first leaf node that is visited.

4) Node Ordering: In a constant radius SD algorithm, the order in which the nodes are visited has no impact on complexity. However, for dynamic radius reduction, if nodes with smaller distances are visited first, the complexity can be reduced significantly. Thus, proper ordering of the nodes is important. There have been many algorithms proposed on the ordering of the nodes. The following are among the most significant ones.

1) Sorted QR

Sorted QR (SQRD) is done in the preprocessing stage. It is a modified version of the QR decomposition,

which sorts the columns so that the PED will be distributed in a manner where efficient pruning can be obtained. SQRD is based on the Gram-Schmidt algorithm. Columns of $\mathbf{H}$, $\mathbf{Q}$, and $\mathbf{R}$ are reordered in each orthogonalization step to minimize the diagonal element of $\mathbf{R}$. This ensures that symbols with large h_i (which most likely have higher SNR) are detected first. The SQRD algorithm is given in Alg. 2.

Input: $\mathbf{H}$

Output: Permutation matrix $\mathbf{P}$ and sorted QR decomposition $\mathbf{Q}$ and $\mathbf{R}$ of $\mathbf{H}$

begin

- $\mathbf{R} \longleftarrow 0$
- $\mathbf{Q} \longleftarrow \mathbf{H}$
- $\mathbf{P} \longleftarrow (1, ..., M_T)$
- **for** $i \leftarrow 1$ **to** M_T **do**
  - $k_i = \arg\min_{l=i,...,M_T} \|q_i\|^2$
  - exchange column i and k_i in $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{P}$
  - $r_{i,i} = \|q_i\|$
  - $\mathbf{q}_i = \mathbf{q}_i / r_{i,i}$
  - **for** $l \leftarrow i+1$ **to** M_T **do**
    - $r_{i,l} = \mathbf{q}_i^H \mathbf{q}_l$
    - $\mathbf{q}_l = \mathbf{q}_l - r_{i,l}\mathbf{q}_l$
  - **end**
- **end**

end

Algorithm 2: The sorted QR algorithm.

2) Schnorr/Euchner Ordering

When the tree is traversed depth first with dynamic radius reduction, if the depth-first strategy is supplemented with the metric-first strategy, SD can be more efficient. This metric-first strategy is known as Schnorr/Euchner (SE) ordering. The basic idea is to give preference to nodes with the smallest PED when choosing the next parent node. By doing so, it is expected that leaf nodes with smaller PEDs will be found earlier which result in rapid shrinkage of the sphere and more efficient pruning of the tree. There have been modified versions of this algorithm that give different weight to the PED depending on the level of the tree. This is to compare the PEDs equally, as PEDs that are closer to the leaf node are expected to be larger. For a real-valued constellation, the SE ordering can be easily obtained by using geometric information. The constellation point that is closest to the received vector will have the smallest PED. The ordering is then given in a zigzag fashion with the received vector as the center point. This is illustrated in Fig. 6 for a 1-dimensional case.

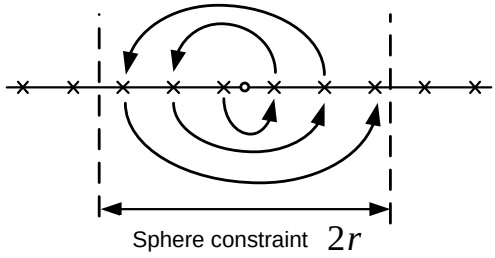


Fig. 6. Schnorr/Euchner ordering for a 1 dimensional constellation.

3) PSK/QAM SE ordering

For a complex valued PSK or QAM constellation, SE ordering can be applied by performing a slight modification. For PSK modulation, the distance from the received signal to a constellation point only depends on the phase difference. This is because for PSK, all the constellation points are equidistant from the origin. Thus, SE ordering can be performed by using the phase in place of the Euclidean distance. For QAM constellations, a combination of PSK-SE ordering and exhaustive search is used. First, constellation points are grouped into sets of equidistant points from the origin. For example, QPSK, 16-QAM, and 64-QAM yield 1, 3, and 9 PSK sets, respectively. Then for each subset, the PSK SE ordering is obtained. Exhaustive search is performed among the minimum distance point of each different PSK subsets to obtain the global minimum.

F. Sub-Optimal Detectors

1) K-best detection: The runtime complexity of SD is variable and depends highly on SNR. This is unfavorable for VLSI implementations as it requires the hardware to be built assuming the worst case complexity, which is equal to an exhaustive ML search. An alternative to SD is the K-Best detector. In K-Best, detection on the tree is performed breadth first, and for each level, only the best K are retained. This gives a constant complexity algorithm, but is now sub-optimal. However, its performance loss is small for sufficiently large K. Simulations show that for a 4×4 16-QAM system, $K = 5$ is a good tradeoff between performance and complexity, which is also verified by [5].

2) Combined ML and DFE Decoding: One technique to reduce the exponential complexity of the SD is to combine linear detectors with optimal detectors to form a reduced complexity sub-optimal detector. In fact, combined ML and DFE decoding, first introduced by [6], is one such technique. As the name implies, this detection scheme combines the low computational complexity of SIC and the optimality of Maximum Likelihood. In its early form, the authors proposed to decode the first p subchannels via ML decoding and use the DFE procedure to decode the remaining subchannels in V-BLAST systems. The key idea of this detection scheme is to not perform a full QR decomposition of $\mathbf{H}$. In other words, the system does not completely upper triangulize the channel matrix as it is

typically done by the ZF-DFE receiver. If we only triangulize $(n-p)$ columns of $\mathbf{H}$ we obtain

$$\mathbf{H} = \tilde{\mathbf{Q}} \begin{pmatrix} \tilde{\mathbf{R}} & \mathbf{H_a} \\ 0 & \mathbf{H_b} \end{pmatrix}$$

where $\mathbf{H_b}$ is a $p \times p$ square matrix, $\mathbf{H_a}$ is an $(n-p) \times p$ matrix, and $\tilde{\mathbf{R}}$ is an $(n-p) \times (n-p)$ upper triangular matrix. Thus, the received signal, after left multiplying by $\tilde{\mathbf{Q}}$, can be written as

$$\tilde{\mathbf{y}} = \begin{pmatrix} \tilde{\mathbf{R}} & \mathbf{H_a} \\ 0 & \mathbf{H_b} \end{pmatrix} \begin{pmatrix} \mathbf{s_a} \\ \mathbf{s_b} \end{pmatrix} + \tilde{\mathbf{v}}$$

Next, the scheme proceeds to jointly detecting the sub-vector $\mathbf{s_b} = [s_{n-p+1}, s_{n-p+2}, ..., s_n]^T$ via a Maximum Likelihood procedure. As SD did not exist at the time this work was presented, the authors used a full blown ML to jointly detect these symbols. However, SD can be used for this step. After this step, the sub-vector $\mathbf{s_a} = [s_1, ..., s_{n-p}]^T$ is detected by first canceling out the interference of $\mathbf{s_b}$ and then using the usual DFE procedure. It is clear that the choice of p has a direct impact on both the complexity and performance of this scheme and this parameter is a key one that could be optimized when designing a combined ML and DFE detector.

3) Reduced Dimension Maximum Likelihood: Reduced dimension ML detector was first introduced in [7]. The idea is to perform ML over a reduced dimension set to reduce the size of the search set. Then use linear techniques to determine the symbols corresponding to the remaining dimension. To reduce the dimension, the channel matrix is vertically divided into two matrices $\mathbf{H}_1$ and $\mathbf{H}_2$ each n_1 and n_2 columns wide ($n_1 + n_2 = M_T$). The ML solution can now be decomposed as

$$\underset{\mathbf{s} \in \mathcal{F}^{M_T}}{\arg\min} \|\mathbf{y} - \mathbf{H}\mathbf{s}\| = \underset{\mathbf{s}_1 \in \mathcal{F}^{n_1}, \mathbf{s}_2 \in \mathcal{F}^{n_2}}{\arg\min} \|\mathbf{y} - \mathbf{H}_2\mathbf{s}_2 - \mathbf{H}_1\mathbf{s}_1\| \tag{9}$$

Which can be further separated as

$$\hat{\mathbf{s}}_1 = \underset{\mathbf{s}_1 \in \mathcal{F}^{n_1}}{\arg\min} \|\mathbf{y} - \mathbf{H}_2 g(\mathbf{y}, \mathbf{s}_1) - \mathbf{H}_1\mathbf{s}_1\| \tag{10}$$

$$g(\mathbf{y}, \mathbf{s}_1) = \underset{\mathbf{s}_2 \in \mathcal{F}^{n_2}}{\arg\min} \|\mathbf{y} - \mathbf{H}_2\mathbf{s}_2 - \mathbf{H}_1\mathbf{s}_1\| \tag{11}$$

By replacing $g(\mathbf{y}, \mathbf{s}_1)$ with a linear estimate

$$\hat{\mathbf{s}}_2 = \mathbf{H}_2^* \left(\mathbf{H}_2\mathbf{H}_2^* + \sigma^2\mathbf{I}\right)^{-1} (\mathbf{y} - \mathbf{H}_1\mathbf{s}_1) \tag{12}$$

and defining a dimension reduction operator $\mathbf{Z} = \sigma^2 \left(\mathbf{H}_2\mathbf{H}_2^* + \sigma^2\mathbf{I}\right)^{-1}$, (10) can be reduced to

$$\hat{\mathbf{s}}_1 = \underset{\mathbf{s}_1 \in \mathcal{F}^{n_1}}{\arg\min} \|\mathbf{Z}\mathbf{y} - \mathbf{Z}\mathbf{H}_1\mathbf{s}_1\| \tag{13}$$

which is the same form as the ML rule and can be efficiently solved by using SD. After $\hat{\mathbf{s}}_1$ has been obtained, $\hat{\mathbf{s}}_2$ can be found by using MMSE detection given in (12). RDML gives a good tradeoff between ML and linear detection by keeping the complexity low while achieving low BER.

III. Soft-Input Soft-Output MIMO Detectors

A. Introduction

Different classes of hard MIMO detectors were described in section II. Specifically, linear and optimal detectors were discussed and it was shown that although linear detectors benefit from the reduced computational complexity, they perform poorly especially at moderate to low values of SNR. In addition, it was also shown that there exists a class of sub-optimal hard detectors which attempts to combine the good features of both linear and optimal detectors. In this section, we will study another category of MIMO detectors commonly known as soft-input soft-output (SISO) MIMO detectors. This category of detectors is exclusively used in MIMO communication systems following the turbo principle [8]. In fact, turbo systems were first introduced by Berrou *et al.* as a promising technique to approach the Shannon capacity for an additive white Gaussian noise (AWGN) channel. This concept was then extended to include MIMO systems. Hence, a new field of *iterative detection and decoding* (IDD) for multiple antenna channels was born [9]. In an IDD receiver, soft information is exchanged between the symbol detector and the channel decoder to achieve performance close to channel capacity. In this section, we will focus on describing several classes of soft symbol detectors for multi antenna channels and keep the discussion on MIMO channel decoders for section IV.

B. Soft Detectors

At the transmitter, a rate R code is used to convert a vector of i.i.d. bits $\mathbf{b}$ to an encoded vector of bits $\mathbf{c}$. While the entries of $\mathbf{b}$ are independent, the entries of $\mathbf{c}$ are clearly dependent. This is why the vector of encoded bits is then multiplexed to different layers (M_T layers) via a serial to parallel converter and interleaved via a random (or deterministic) interleaving matrix. Hence, we obtain M_T vectors of encoded bits $\{\mathbf{c_i}\}_{i=1}^{M_T}$ that are almost independent due to interleaving. In the sequel, we will assume that the encoded and reshuffled bits are independent. The last stage at the transmitter is the modulation stage. The modulation format used on the LTE-A uplink is single carrier frequency domain multiple access (SC-FDMA). At the receiver, the first two stages are down conversion and demodulation. Since LTE-A systems use SC-FDMA for modulation, the demodulation procedure should be be tailored to this modulation technique. After down-conversion and demodulation, we know, from section II, that the received signal vector is given by $\mathbf{y} = \mathbf{Hs} + \mathbf{v}$. Assuming that each symbol s_k belongs to a 2^q-ary QAM alphabet, we label the bits associated with s_k by $c_{k,1}, c_{k,2}, ..., c_{k,q}$. Thus, the symbol vector $\mathbf{s}$ comprises of qM_T bits. The task of the soft symbol detector is to compute the *a posteriori log-likelihood ratio* (APLLR) for each bit. Note that the received symbol vector constitutes of qM_T bits. Assuming a spatially additive white Gaussian noise, the APLLR is defined as and given by

$$L_{post}(c_{k,i}) = \log \frac{P(c_{k,i} = +1|\mathbf{y})}{P(c_{k,i} = +1|\mathbf{y})} = \log \frac{\sum_{\mathbf{s}\in\mathcal{S}_{k,i}^{+}} e^{-d(\mathbf{s})/\sigma}}{\sum_{\mathbf{s}\in\mathcal{S}_{k,i}^{-}} e^{-d(\mathbf{s})/\sigma}} \tag{14}$$

where

$$d(\mathbf{s}) = ||\mathbf{y} - \mathbf{Hs}||^2 - \sigma \sum_{i=1}^{M_T} \sum_{j=1}^{q} \log P(c_{i,j})$$

$$P(c_{i,j}) = \frac{1}{2}(1 + c_{i,j}\tanh(\frac{L_{pri}(c_{i,j})}{2}))$$

The set $\mathcal{S}_{k,i}^{+}$ is the set of all symbols satisfying $c_{k,j} = +1$ ($\mathcal{S}_{k,i}^{-}$ is defined similarly), and $L_{pri}(c_{k,i})$ is the *a priori log-likelihood ratio* defined as

$$L_{pri}(c_{k,i}) = \log \frac{P(c_{k,i} = +1)}{P(c_{k,i} = -1)} \tag{15}$$

The symbol detector receives a symbol vector $\mathbf{y}$, computes the APLLRs assuming equal priors on all transmitted bits $c_{k,i}$, and passes the soft information to the channel decoder which in turn computes the new priors and feeds them back to the symbol detector. This process is iterated until a certain convergence criterion is met. At that point, the ALLRs are quantized and consequently, hard decisions are made.

C. Soft Sphere and Tree based Detectors

After outlining the basic role of a soft MIMO detector, we now focus on discussing different implementations of such detector. Ultimately, we are seeking a device that, given a received vector $\mathbf{y}$ and *a priori likelihood ratios*, outputs an updated version of the *a posteriori likelihood ratios* (ALLRs). One way to solve for the ALLRs is to derive a soft-input soft-output detector that solves for the exact ALRRSs via a brute force search over all $\mathbf{s} \in \mathcal{F}^{M_T}$. The task of this detector is to compute

$$L_{post}(c_{k,i}) = \log \frac{\sum_{\mathbf{s}\in\mathcal{S}_{k,i}^{+}} e^{-d(\mathbf{s})/\sigma}}{\sum_{\mathbf{s}\in\mathcal{S}_{k,i}^{-}} e^{-d(\mathbf{s})/\sigma}}$$

for each bit. This computation, however, requires searching over all $\mathbf{s} \in \mathcal{F}^{M_T}$ to obtain the sets $\mathcal{S}_{k,i}^{+}$ and $\mathcal{S}_{k,i}^{-}$ and then marginalizing the a posteriori probabilities over these sets. This, of course, is computationally prohibitive and not implementable for large block codes and/or large signal constellations. In this section, we show how the hard sphere decoder, which finds the hard ML estimate using tree based search algorithms, can be altered to output the ALLRs of the bits comprising the received vector. First we recall, from section II-E, that a sphere decoder computes the following path metric

$$p(\mathbf{s}) = \|\tilde{\mathbf{y}} - \mathbf{R}\mathbf{s}\| = \sum_{i=1}^{M_T} \left| \tilde{y}_i - \sum_{l=i}^{M_T} r_{i,l} s_l \right|^2 \tag{16}$$

for a collection of symbols living inside a sphere of radius B via a breadth first search or depth first search algorithm. Afterwards, the decoder chooses the symbol vector with the smallest path metric as the quantized (hard) decision. While this procedure requires less computations relative to the full hard ML detector it still suffers from an expected exponential computational complexity. In order to derive the soft sphere detector we start by obtaining a relationship between $d(\mathbf{s})$ and $p(\mathbf{s})$. If we recompute $d(\mathbf{s})$ while taking the QR factorization of the channel matrix $\mathbf{H}$ into account

$$d(\mathbf{s}) = ||\tilde{\mathbf{y}} - \mathbf{R}\mathbf{s}||^2 - \sigma \sum_{i=1}^{M_T}\sum_{j=1}^{q} \log P(c_{i,j}) = \sum_{i=1}^{M_T} \left(p(\mathbf{s}^i) - \sigma \sum_{j=1}^{q} \log P(c_{i,i}) \right) + C$$

where $p(\mathbf{s}^i) = \left|\tilde{y}_i - \sum_{l=i}^{M_T} r_{i,l} s_l\right|^2$ and $\mathbf{s}^i = [s_i, s_{i+1}, ..., s_{M_T}]^T$ is the partial symbol vectors (PSVs) as in [10]. Note that since the constant term C appears in $d(\mathbf{s})$ for all $\mathbf{s}$ we can drop it with out loss of optimality. In addition, since we are summing over the PSVs, they can be arranged in a tree that has its root just above level $i = M_T$ and leaves, on level $i = 1$, which correspond to symbol vectors $\mathbf{s}$. The Euclidean distances $d(\mathbf{s})$ can be computed recursively as $d(\mathbf{s}) = d_1$ with the partial Euclidean distances (PEDs) $d_i = d_{i+1} + |e_i|^2$, $i = M_T, M_T - 1, ..., 1$, where $d_{M_T+1} = 0$ and the distance increments (DIs)

$$|e_i|^2 = |\tilde{y} - \sum_{j=i}^{M_T} r_{i,j} s_j|^2 \tag{17}$$

Since the dependence of the PEDs d_i on the symbol vector $\mathbf{s}$ is only through the PSV $\mathbf{s}^i$, we have transformed the soft ML detection and the computation of the ALLRs into a weighted tree-search problem: PSVs and PEDs are associated with nodes, branches correspond to DIs. Using $d(\mathbf{s})$, the tree detection algorithm compares the reliability of distinctive paths and chooses the surviving paths. In other words, the tree detection algorithm finds the complete paths associated with smallest $d(\mathbf{s})$. Denoting the set of the corresponding symbol candidates as $\mathcal{L}$, the approximate ALLRs can be expressed as

$$L_{post}(c_{k,i}) \approx \log \frac{\sum_{\mathbf{s} \in \mathcal{L} \cap \mathcal{S}_{k,i}^+} e^{-d(\mathbf{s})/\sigma}}{\sum_{\mathbf{s} \in \mathcal{L} \cap \mathcal{S}_{k,i}^-} e^{-d(\mathbf{s})/\sigma}} \tag{18}$$

We can also use the max-log approximation to further simplify the above approximation

$$L_{post}(c_{k,i}) \approx \left(\min_{\mathbf{s} \in \mathcal{L} \cap \mathcal{S}_{k,i}^+} d(\mathbf{s}) - \min_{\mathbf{s} \in \mathcal{L} \cap \mathcal{S}_{k,i}^-} d(\mathbf{s}) \right) \tag{19}$$

Although soft sphere detectors and tree based detectors are less complex relative to a soft ML detector they still have an exponential computational complexity which is not desirable for large number of antennas and/or large signal signal constellations. This is why, we will next study some simpler sub-optimal techniques that achieve reduced complexity. Finally, it is important to mention that in the case when there is no passing of soft information back and forth between the channel detector and the channel decoder, the soft sphere decoder assumes equal priors on bits and the max-log approximation becomes

$$L_{post}(c_{k,i}) \approx \left(\min_{\mathbf{s} \in \mathcal{L} \cap \mathcal{S}_{k,i}^+} ||\tilde{\mathbf{y}} - \mathbf{R}\mathbf{s}||^2 - \min_{\mathbf{s} \in \mathcal{L} \cap \mathcal{S}_{k,i}^-} ||\tilde{\mathbf{y}} - \mathbf{R}\mathbf{s}||^2 \right) \tag{20}$$

D. Soft MMSE Detector

In order to reduce the computational burden imposed by a soft ML or sphere decoder, a soft MMSE detector could be used [11]. As we already know, the task of the soft channel detector is to calculate the ALLRs given the a priori distribution of the transmitted bits and the received signal vector. Instead of solving for the exact ALLRs, one could use a linear MMSE detector to first estimate the transmitted symbols and then compute the ALLRs based on this estimate. We start deriving the soft MMSE by rewriting the received signal as

$$\mathbf{y} = \mathbf{h}_k s_k + \mathbf{H}_k \mathbf{s}_k + \mathbf{n} \tag{21}$$

where $\mathbf{H}_k = [\mathbf{h}_1\mathbf{h}_2, ..., \mathbf{h}_{k-1}, \mathbf{h}_{k+1}, ..., \mathbf{h}_{M_r}]$, and $\mathbf{s}_k = [\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_{k-1}, \mathbf{s}_{k+1}, ..., \mathbf{s}_{M_r}]$. The decision statistic of the k^{th} substream using a linear filter $\mathbf{w}_k$ is

$$\tilde{y}_k = \mathbf{w}_k^*\mathbf{y} = d_k + u_k + v_k \tag{22}$$

where $d_k = \mathbf{w}_k^*\mathbf{h}_k s_k$ is the desired response obtained by linear beamforming, $u_k = \mathbf{w}_k^*\mathbf{H}_k\mathbf{s}_k$ is the co-antenna interference (CAI), and $v_k = \mathbf{w}_k^*\mathbf{v}$ is the phase rotated noise respectively. In this scheme, the CAI is removed from the linear beamformer output to obtain $x_k = \tilde{y}_k - u_k$ where u_k is the linear combination of interfering substreams and x_k is the improved estimate of transmitted symbol s_k. As proposed in [11], the multisubstream detector optimizes the interference estimate and the weights of the linear detector jointly. The performance of the estimator is measure by the error $e_k = s_k - x_k$. The weights $\mathbf{w}_k$ and the interference estimate u_k are optimized by minimizing the mean-square value of the error between each substream and its estimate

$$(\hat{\mathbf{w}}_k, \hat{u}_k) = \arg\min_{\mathbf{w}_k, u_k} \mathbb{E}[||s_k - x_k||^2] \tag{23}$$

where the expectation $\mathbb{E}$ is taken over the noise and the statistics of the data sequence. The solution to the problem is given by

$$\hat{\mathbf{w}}_k = (\mathbf{P} + \mathbf{Q} + \Sigma_{M_R})^{-1}\mathbf{h}_k \tag{24}$$

$$\hat{u}_k = \mathbf{w}_k^*\mathbf{z} \tag{25}$$

where

$$\mathbf{P} = \mathbf{h}_k\mathbf{h}_k^*$$

$$\mathbf{Q} = \mathbf{H}_k[\mathbf{I}_{(M_R-1)} - diag(\mathbb{E}[\mathbf{s}_k]\mathbb{E}[\mathbf{s}_k]^*)]\mathbf{H}_k^*$$

$$\Sigma_{M_R} = \sigma^2\mathbf{I}_{M_R}$$

$$\mathbf{z} = \mathbf{H}_k\mathbb{E}[\mathbf{s}_k]$$

In solving for $\mathbb{E}[\mathbf{s}_k]$, the relationship between the expectation and the a priori distributions is given by

$$\mathbb{E}[c_{k,j}] = \tanh\left(\frac{L_{pri}(c_{k,j})}{2}\right),\ j = 1, 2, ..., B$$

In arriving to the above solution, it was assumed that $\mathbb{E}[\mathbf{v}\mathbf{v}^*] = \sigma^2\mathbf{I}_{M_R}$, $\mathbb{E}[\mathbf{s}\mathbf{v}] = \mathbf{0}$, and $\mathbb{E}[s_i s_j] = \mathbb{E}[s_i]\mathbb{E}[s_j]$ $\forall i \neq j$. These conditions are achieved via independent and different space and time interleaving at the transmitter. For the first iteration, it is assumed that $\mathbb{E}[\mathbf{s}_k] = \mathbf{0}$, in which case x_k is given by

$$x_k = \mathbf{h}_k^*(\mathbf{H}\mathbf{H}^* + \sigma^2\mathbf{I})^{-1}\mathbf{y}$$

With increasing number of iterations, we assume that in the limit, $\mathbb{E}[\mathbf{s}_k] \to \mathbf{s}_k$, in which case, x_k simplifies to

$$x_k = (\mathbf{h}_k^*\mathbf{h}_k + \sigma^2)^{-1}\sigma_k^*(\mathbf{y} - \mathbf{H}_k\mathbf{s}_k)$$

due to the perfect soft interference canceler. Finally, to acquire the expectations of interfering substreams, we use the a priori distributions of the transmitted bit streams provided by the channel decoder at the previous iteration.

IV. Channel Decoders

A. Convolutional Codes

1) Encoding: Convolutional codes are encoded by using a linear feedback shift register (LFSR). A convolutional code is defined by the number of memory elements and the topology of the feedback and feedforward network, which is represented in compact form by the generator polynomial. A simple convolutional encoder is shown in Fig. 7.

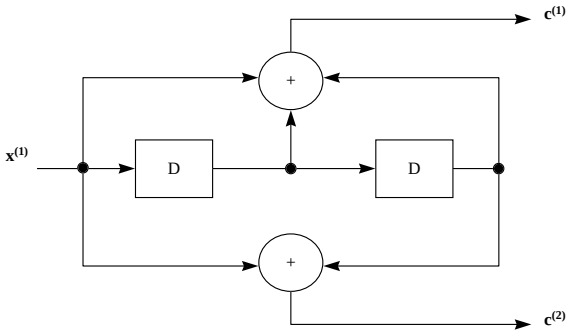


Fig. 7. A simple convolutional encoder with memory size 2, and rate 1/2.

2) Decoding: Several algorithms exist for decoding convolutional codes. For relatively small values of k, the Viterbi algorithm is universally used as it provides maximum likelihood performance and is highly parallelizable. Viterbi decoders are thus easy to implement in VLSI hardware and in software on CPUs with SIMD instruction sets.

Longer constraint length codes are more practically decoded with any of several sequential decoding algorithms, of which the Fano algorithm is the best known. Unlike Viterbi decoding, sequential decoding is not maximum likelihood but its complexity increases only slightly with constraint length, allowing the use of strong, long-constraint-length codes.

Both Viterbi and sequential decoding algorithms return hard-decisions: the bits that form the most likely codeword. An approximate confidence measure can be added to each bit by use of the soft output Viterbi algorithm. Maximum a posteriori (MAP) soft-decisions for each bit can be obtained by use of the BCJR algorithm.

B. Turbo Codes

Turbo codes were first introduced in 1993 by Berrou, Glavieux, and Thitimajshima [12], and were among the first codes to achieve near capacity.

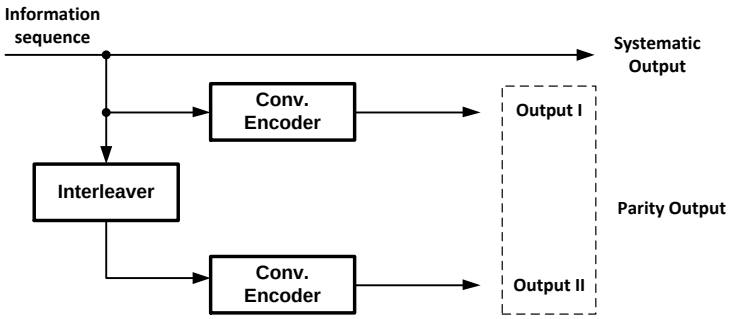


Fig. 8. A turbo encoder (parallel concatenated convolutional code).

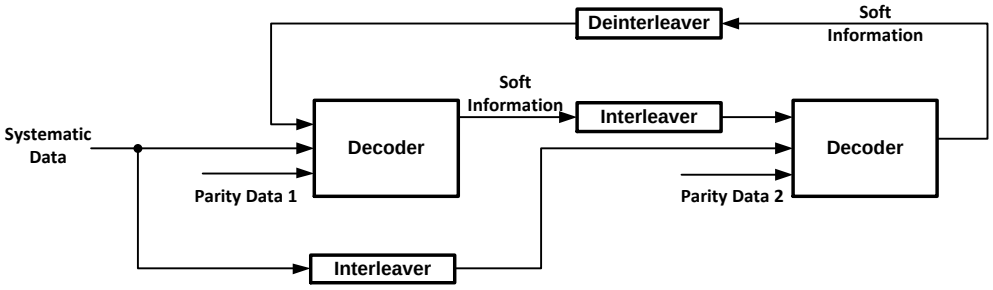


Fig. 9. A turbo decoder. Exchange of soft information is the key to its high performance.

1) Encoding: Turbo codes are constructed as a combination of two or more component codes on different interleaved version of the same information sequence. Turbo encoding is usually done by encoding the information sequence using a convolutional code, then interleaving the sequence, and finally performing another convolutional encoding on the interleaved sequence. The first convolutional code consists the outer code, while the second code is the inner code. This process is depicted in Fig. 8

2) Decoding: Turbo decoding is performed by using an iterative decoder. A high level block diagram is given in Fig. 9. For each constituent code, a separate decoder exists. Each decoder outputs soft information of the decoded bits. The other decoder uses this soft information as its *a posteriori probability*. Decoding continues for a number of iterations until the outputs converge, or a certain criterion is met.

V. Iterative MIMO Receiver

In an iterative turbo MIMO receiver, the main concept is the same as that of the turbo decoder shown in Fig. 9. Turbo principles can be applied to any system with a channel code as the outer code, while the inner code is considered to be the channel effects. This is true for turbo equalizers where the inner code of the serial concatenated turbo code is assumed to be the inter-symbol-interference (ISI) channel. Here the inner code is defined as a rate 1 code over the field of real or complex numbers. The concept of such an iterative turbo equalizer has been demonstrated

and shows perfect equalization for a multipath Gaussian channel [13].

Extending the turbo principle to MIMO detection, the spatial multiplexing of a MIMO channel can be viewed as the inner code, while there exists an outer code, a turbo code in the case of LTE-A systems. Thus, the crux of a turbo MIMO receiver lies in the development of a soft-input soft-output MIMO detector, along with an efficient iterative structure that can pass information between the channel decoder and the MIMO detector. The fundamental structure of such a receiver is shown in Fig. 10.

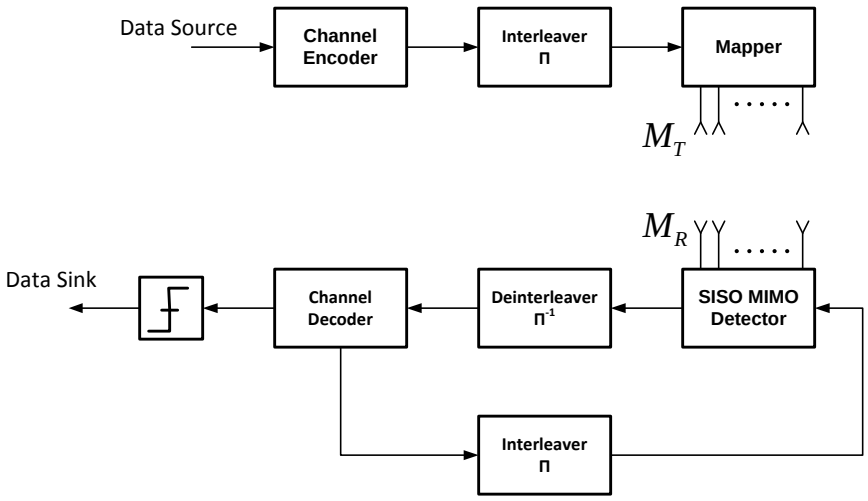


Fig. 10. Fundamental architecture of an iterative turbo MIMO receiver.

A. Turbo BLAST

One of the first work on turbo MIMO receiver is the work done by Sellathurai and Haykin [11]. In this work, Bell labs layered space time (BLAST) is extended and casted in the context of turbo decoding. Foschini has answered the fundamental question of constructing a BLAST system whose capacity grows linearly with the number of transmit antennas. D-BLAST was the solution to this problem. However, D-BLAST is too complex for any practical implementation. Turbo-BLAST plays an important role in reducing the complexity of D-BLAST for practical implementations.

In turbo-BLAST, a random space-time block (RSTB) code is used. RSTB is implemented by first encoding substreams using independent FEC codes. A following space-time random interleaver is used to create an inter-stream permutation. The space-time interleaver first interleaves in time, then permutes across space. The space interleaver ensures the spatial cycling of each substream over all possible sub channels (transmit antennas).

In theory, the RSTB code can be modeled as a single Markov process, and a trellis can be formed that includes the effect of interleaving. However, such a trellis representation is extremely complex and no feasible decoding is possible. Borrowing from turbo principles, the FEC code is considered as an outer code, while the wireless channel (a time varying matrix channel) is viewed as an inner code. This is depicted in Fig. 11.

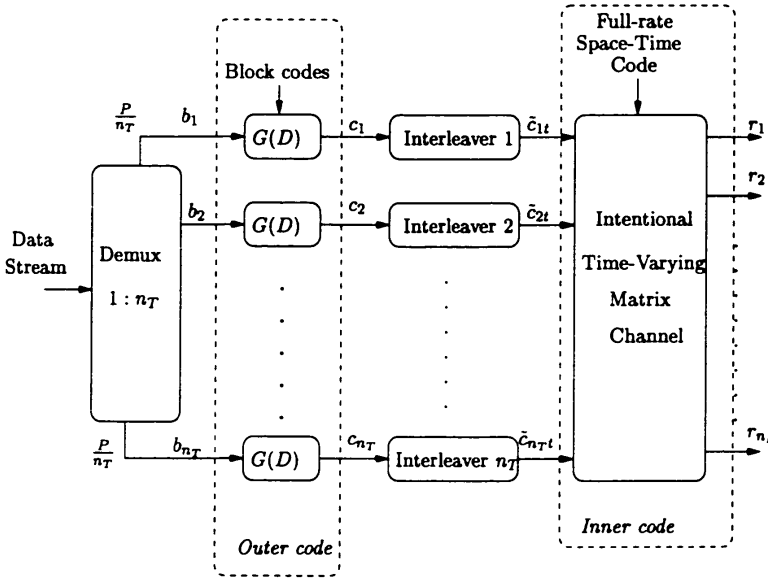


Fig. 11. RSTB codes as a serially concatenated code. Courtesy of Sellathurai et al. [11].

In an iterative decoding scheme, the two decoding stages will be a SISO detector for the inner code, and a set of parallel SISO channel decoders for the outer code. The two stages are separated by the space-time interleaver/deinterleaver. This process is depicted in Fig. 12.

As the soft detector, several choice can be used. The optimal detector is the MAP detector, where the probability of the k^{th} substream is obtained by averaging out the contributions of the interfering substreams. As is with any MAP scheme, the complexity of the soft MAP detector is large, and exponential with the number of transmit antennas. However, a sub-optimal scheme that employs the max-log approximation can be used.

Any standard SISO decoder for the employed FEC code can be used as the outer decoder. The performance of the turbo-BLAST system is shown in Fig. 13.

B. MMSE Turbo Receiver

A different frequency-selective fading channel has been considered in [14]. In this paper, coding is introduced in space and frequency dimension by means of bit-interleaved coded modulation (BICM). To perform the MIMO space

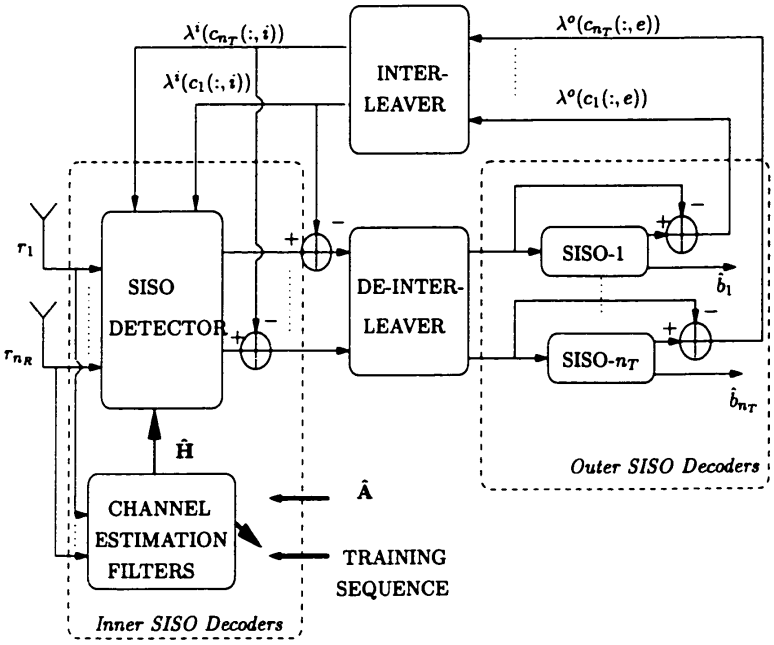


Fig. 12. Iterative decoder for turbo-BLAST. Courtesy of Sellathurai et al. [11].

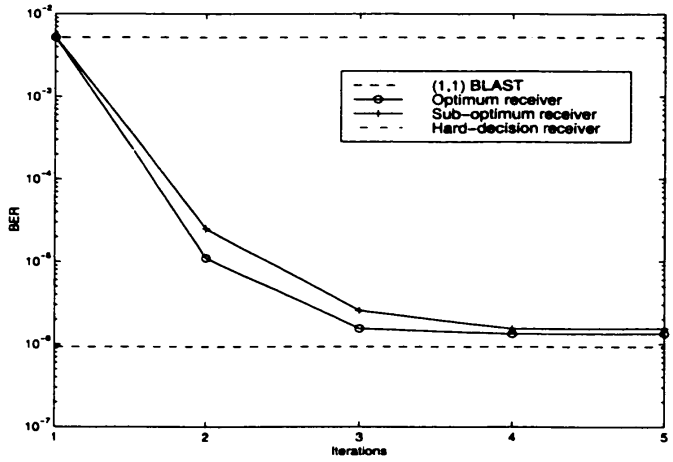


Fig. 13. Performance of Turbo-BLAST. Courtesy of Sellathurai et al. [11].

equalization, a set of SISO MMSE equalizers are used which perform demodulation as well. The soft information is exchanged between the SISO MMSE equalizer and the channel decoder in a turbo principle. Overall, OFDM is used to convert the frequency selective channel to several flat fading channels. The basic block diagram of the transmitter is given in Fig. 14.

The iterative receiver employs the turbo principles by using a SISO outer decoder and a soft output space equal-

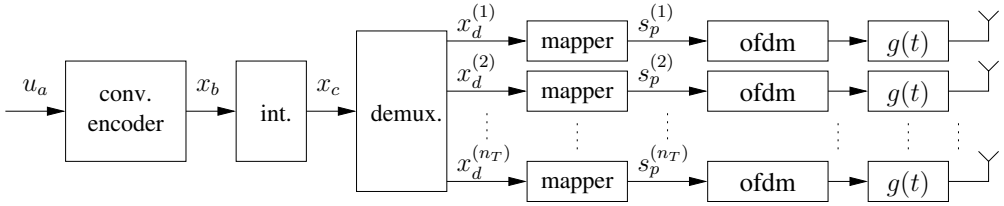


Fig. 14. Transmitter for MMSE turbo. Courtesy of Zuyderhoff et al. [14].

izer/demodulator which is separated by an interleaver and deinterleaver as depicted in Fig. 15. The outer decoder simply implements the SISO convolutional decoder based on the BCJR algorithm. The inner SISO equalizer needs to mitigate the spatial interference as well as properly demodulate the symbols. By assuming the independence of the interleaved and demultiplexed coded bits, the two stages can be split as shown in Fig. 16.

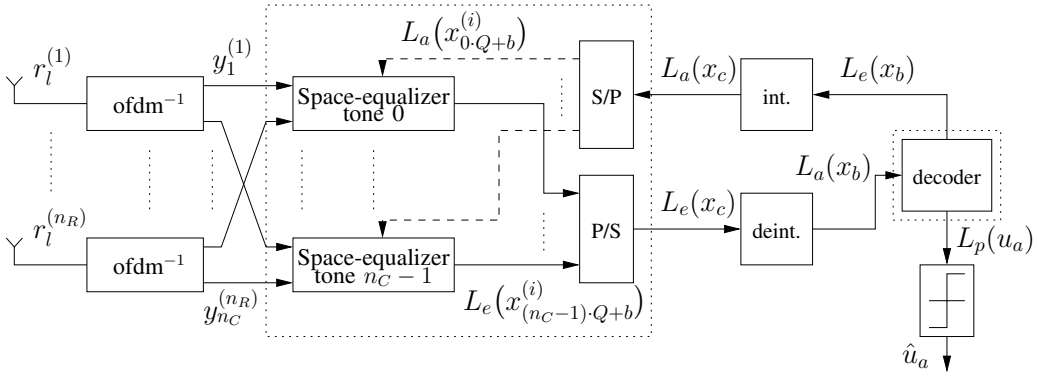


Fig. 15. Receiver for MMSE turbo.

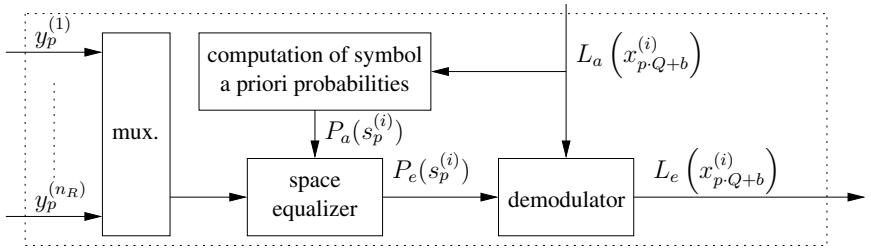


Fig. 16. Space equalizer for MMSE turbo. Courtesy of Zuyderhoff et al. [14].

For the equalization, a low complexity MMMSE solution is derived based on the optimal trellis approach of turbo

equalization. A linear block operation is used to produce the estimated transmitted symbol without relying on the a priori information to keep the extrinsic information independent. Performance of the MMSE turbo receiver is shown in Fig. 17 which also includes comparison of the MMSE approximation to the optimal MAP.

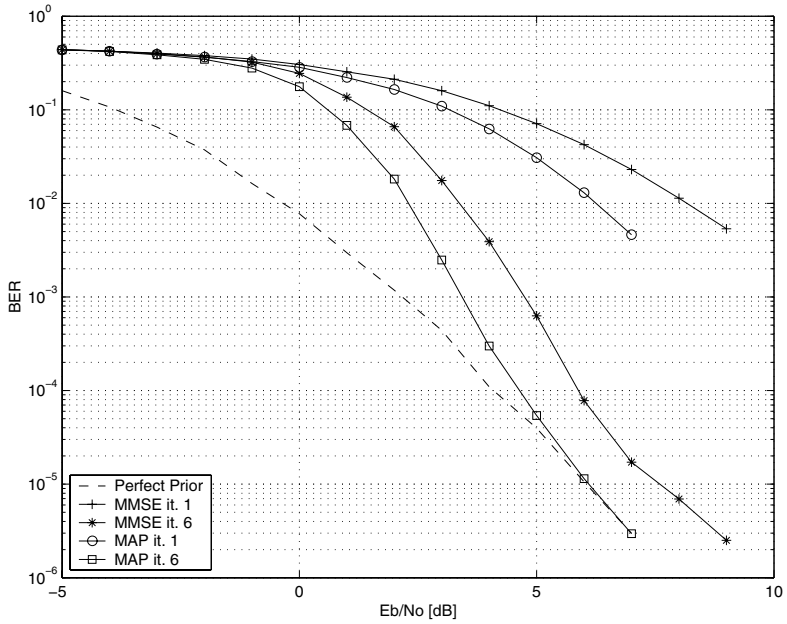


Fig. 17. Performance of MMSE turbo receiver. Courtesy of Zuyderhoff et al. [14].

C. SIC Turbo Receiver

Wang and Li have proposed the use of a probability data association (PDA) SIC detector [15]. This detector differs from the pervious MMSE based SIC detector in many aspects. Its block diagram is depicted in Fig. 18. There are three parts: soft symbol estimator, soft interference cancellation block and an extrinsic bit LLR calculator. The soft symbol estimator takes the a priori information $L(b)$ and estimates of the transmitted signals. Once the estimate is obtained, the soft interference cancellation block cancels out the effect of these symbols. Finally, the extrinsic information is calculated for the channel decoding block. While calculating the LLRs, a Gaussian forcing technique is employed where the symbols are viewed as a Gaussian random variable.

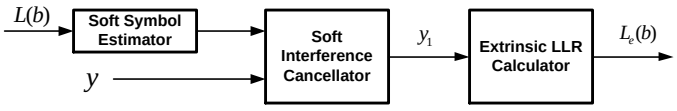
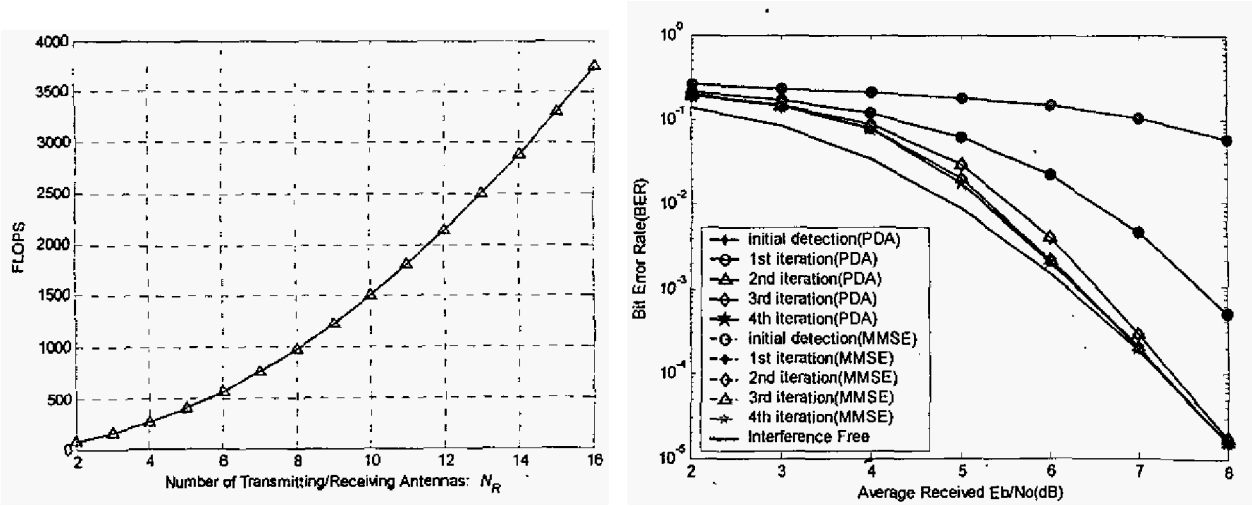


Fig. 18. SISO PDA SIC MIMO detector. Reproduced courtesy of Wang et al. [15].

The main difference between PDA SIC detector and MMSE SIC detector is in the calculation of the likelihood function for the transmitted symbol, where the PDA approach has much less complexity. Also, the intermediate computation in the PDA SIC detector uses matrices with lower condition number, which results in a more numerically stable algorithm. The comparison of PDA SIC to MMSE SIC is shown if Fig. 19. The reduction in complexity with the number of receive antennas and the BER performance are also shown.



(a) (b)

Fig. 19. Comparison of a PDA SIC MIMO detector to a MMSE SIC MIMO detector: (a) reduced FLOPS for PDA SIC, and (b) BER performance. Courtesy of Wang et al. [15].

D. Tree-based Turbo Receiver

In [16], a soft sphere detector has been combined with several channel codes such as convolutional, turbo and LDPC codes. The standard turbo receiver loop has been employed to construct an iterative turbo MIMO receiver. The performance compared to an ML detector is shown in Fig. 20

In [17], an iterative MIMO receiver is implemented using a soft K-best detector. The soft K-best detector employs the same principles as with the soft sphere detector. The difference is the process in selecting the symbols that contribute to the soft information, which is based on the K-best algorithm. The outer code used is a space-time BICM. The performance comparison to a list sphere decoder is shown in Fig. 21

The tree based soft detectors have been implemented in hardware by Chen and Zhang [18]. By using three approximation methods, (i) bidirectional partial tree search, (ii) hybrid two-step detection, and (iii) ECC feedback aided detection bypassing, the authors showed a practical implementation of a soft K-best detector. In partial tree search, only the first few dimensions are fully searched by the breadth first search algorithm. The remaining

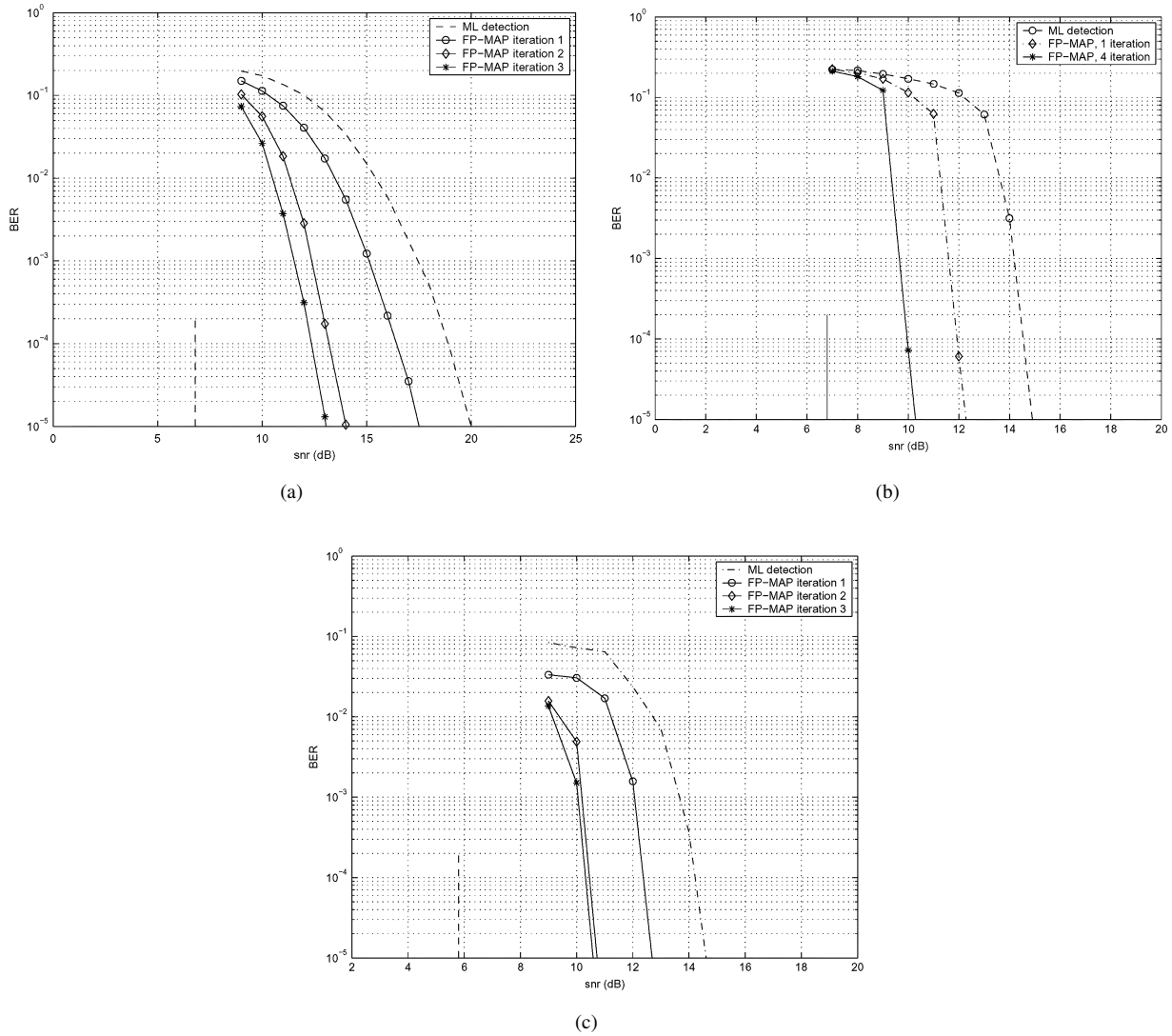


Fig. 20. Performance of iterative MIMO receiver utilizing a soft sphere detector: BER performance of 4×4 system employing (a) rate 1/2, 9216-bits-long convolutional code, 16-QAM, (b) rate 1/2 turbo code, 16-QAM, and (c) rate 8/9 LDPC code, 4-QAM. Courtesy of Vikalo et al. [16].

dimensions are then chosen by the nearest neighbor rule. This process is depicted in Fig. 22.

This can be thought of a simplification of RDML. In addition to that, another tree with the leaf node of the original tree as the root is constructed and the same partial tree search is conducted. The two searches combined together give rise to a bidirectional partial tree search. The candidates obtained through this bidirectional search are then used to compute the soft information.

In hybrid two-step detection, the K-best detection stage is optimized along with the channel decoder. The K-best

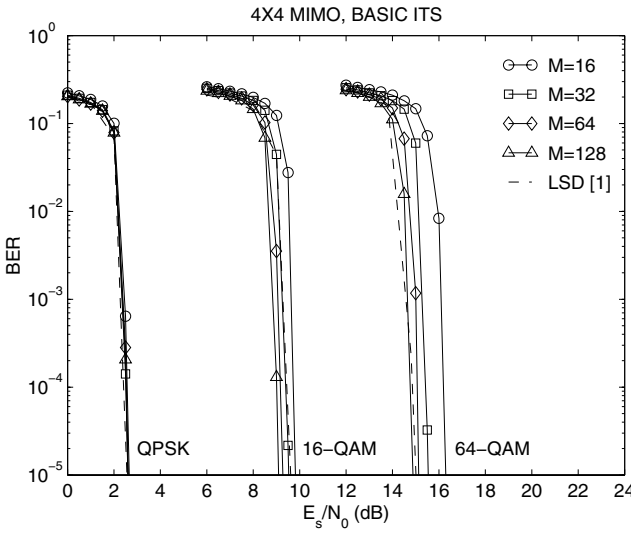


Fig. 21. Error performance of an ST-BICM MIMO system employing the ITS detector, for different list sizes M. Courtesy of de Jong et al. [17]

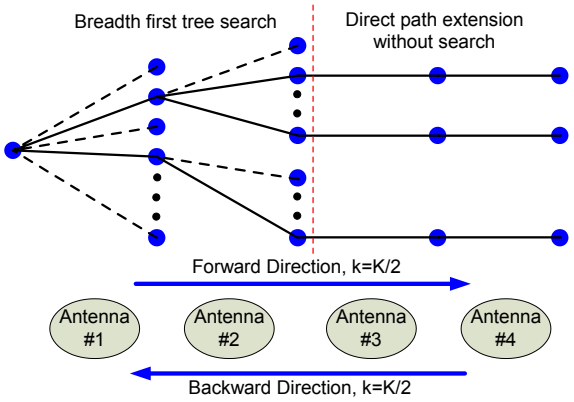


Fig. 22. Bidirectional partial tree search diagram. Courtesy of Chen et al. [18].

stage is performed with a low value of K, which has low complexity. The channel decoding is performed on this approximate output. If the channel decoder fails, then a higher order of K is used which gives near ML performance. This process is depicted in Fig. 23.

The ECC feedback aided detection simplifies the high order K-best detection stage when the first low order K-best detection has caused the channel decoder to fail. By feeding back the soft information obtained from the channel decoder, the detection can be done more accurately, and also the bits that have sufficiently correct soft information from the low order K-best stage need not be detected again.

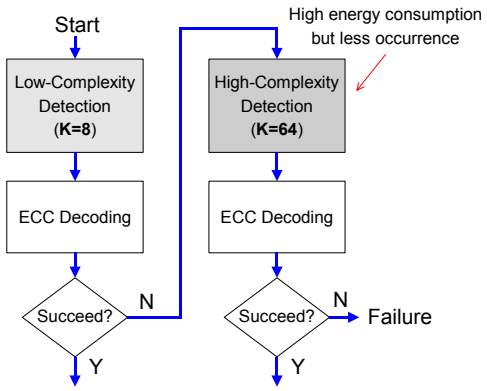


Fig. 23. The flow diagram of the hybrid two-step detection. Courtesy of Chen et al. [18].

By using these approximation techniques, the MIMO receiver was able to achieve near-optimum detection performance, while supporting over 100Mbps throughput with only $0.24mm^2$ silicon area and 11mw power, leading to a $\times 10$ improvement over the state of the art existing systems.

E. Complexity comparison of LTE MIMO receivers

As seen in the previous section, a broad range of iterative MIMO receivers exist. In this section, complexity and performance comparison of an MMSE receiver, a K-best receiver and their iterative counterpart is given. In [19], FPGA implementations as well as ASIC implementation of MIMO receivers have been studied in the context of LTE. 8 iterations were used for turbo decoding. Figure 24 shows the throughput for a linear MMSE, SIC and K-best receivers with the typical urban (TU) correlated channel and an uncorrelated channel.

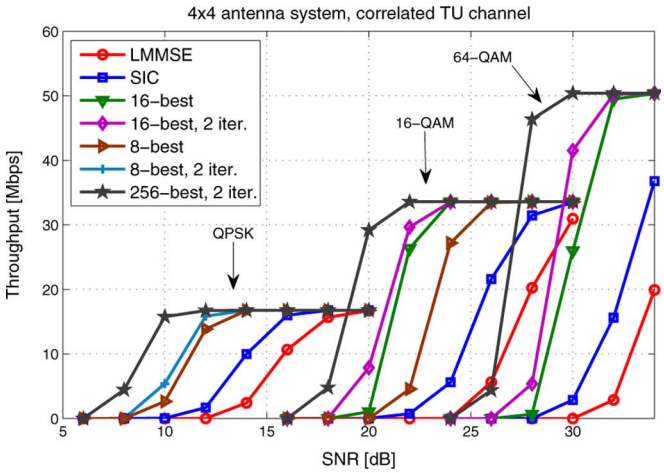
Theoretical computation complexity is presented in Table 1.

SGR denotes the squared Givens rotation, while Gram Schmidt was used for the QR decomposition (QRD). Using an ANSI C++ synthesis tool, catapult C Synthesis tool [20], each receiver algorithm was synthesized. Complexity comparison of a 2×2 system and a 4×4 system is given in Table 2.

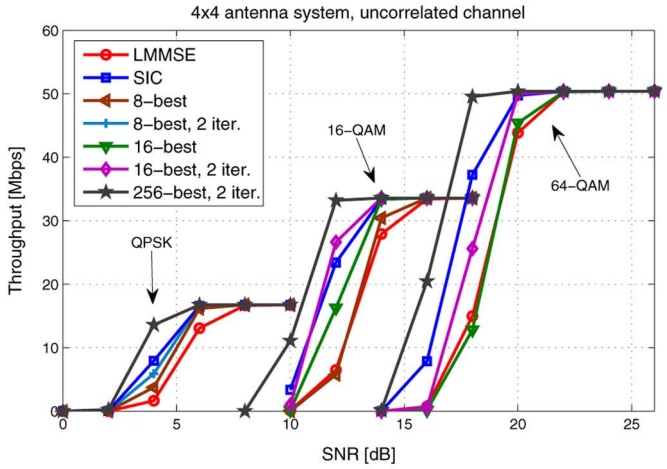
The lowest complexity receiver is dependent on the SNR and channel model. Table 3 tabulates the best receiver for several SNR and different correlation in the channel. A reconfigurable receiver which can adapt to the channel conditions would achieve the best performance, while consuming the least power.

VI. Proposed Work

This section outlines specific system and architectural exploration to be undertaken for this project.



(a)



(b)

Fig. 24. Throughput of MIMO receivers in (a) a typical urban correlated channel, and (b) an uncorrelated channel. Courtesy of Ketonen et al. [19].

Table 1. The theoretical complexity of the receivers as numbers of arithmetic operations. Courtesy of Ketonen et al. [19].

Block	**Multiplications**	**Additions**	**Comparisons**	**Memory**
$(\mathbf{H}^{\mathrm{H}}\mathbf{H}+\sigma^2\mathbf{I}_M)$	$4M^3$	$(4M-2)M^2+M$		$16M^2$
SGR $(\mathbf{Q},\mathbf{R})$	$4M^3$+$4.5M^2$+$7.5M$-12	$4M^3+35M^2-45M-8$		$71M^2+18M$
$\mathbf{R}$ inverse	$\sum_{j=1}^{M-1} j^2$	$\sum_{j=1}^{M-1}(j^2-j)+40M$		$20M^2$
$\mathbf{R^{-1}QH}$	$8M^3$	$(8M-4)M^2$		$32M^2$
Detection	$4M^2$	$3M^2-M$		$16M$
LLR (MMSE/SIC)	[2|8|16]M	$MQ/4$	[2|8|20]M	$8MQ$
Expectation calc.	$4M$		$M(18+Q/2)$	$8M$
SIC	$2M^2+4M$	$4M^2-M$		$80M$
QRD	$8M^3-M$	$8M^3-2M^2+93M$		$32M^2+40M$
Matrix-vector mult.	$4M^2$	$4M^2-2M$		$32M$
K-best LSD	$\sum_{l=2}^{2M} K\sqrt{\Omega}(l+1)$	$K\sum_{l=2}^{2M}\sqrt{\Omega}(l+1)$	$\sum_{l=2}^{2M} K^2\sqrt{\Omega}(l$+1)	$16K(M\sqrt{\Omega}$+$\sqrt{\Omega}$+M+1)
LLR (K-best)	K	$QM+32$	KQM	$5MQ$

Table 2. Complexity comparison of MIMO receivers in (a) a 2×2, and (b) a 4×4 system. Courtesy of Ketonen et al. [19].

Receiver	**Complexity** (GE)	**Power** mW	**Goodput** at 16 dB	**Goodput** at 20 dB
LMMSE	58 k	54	0.3 Mb/s	15.7 Mb/s
SIC	85 k	83	6.2 Mb/s	49.4 Mb/s
8-best	197 k	175	17.4 Mb/s	67.3 Mb/s
8-best, 2 iter.	236 k	251	32.4 Mb/s	69.8 Mb/s

(a)

Receiver	**Complexity** (GE)	**Power** mW	**Goodput** at 24 dB	**Goodput** at 28 dB
LMMSE	480 k	376	0.1 Mb/s	84.4 Mb/s
SIC	540 k	449	20.6 Mb/s	133.7 Mb/s
8-best	582 k	568	113.4 Mb/s	140 Mb/s
8-best, 2 iter.	634 k	700	128 Mb/s	140 Mb/s

(b)

Table 3. Receiver with the best goodput (QAM constellation). Courtesy of Ketonen et al. [19].

SNR	**Corr.**	**Mod. corr.**	**Uncorr.**
	2×2 system		
2 dB	it. 16-best (4)	it. 16-best (4)	it. 16-best (4)
10 dB	16-best (4)	16-best (4)	LMMSE (4)
15 dB	SIC (4)	SIC (16)	SIC (16)
20 dB	it. 8-best (16)	SIC (64)	SIC (64)
	4×4 system		
10 dB	it. 8-best (4)	8-best (4)	LMMSE (4)
15 dB	8-best (4)	LMMSE (4)	SIC (16)
20 dB	LMMSE (4)	8-best (16)	SIC (64)
25 dB	it. 8-best (16)	LMMSE (16)	LMMSE (64)

A. Turbo scheduling

Turbo MIMO receivers consisting of an inner turbo decoder loop and an outer loop between a SISO MIMO detector and the turbo decoder, offers the possibility of BER-optimally allocating the number of iterations between the two loops given a fixed iteration budget. For an iterative turbo MIMO receiver, where the channel decoder itself is also iterative (as in turbo or LDPC code), it is not clear whether it is beneficial to run the iterative channel decoder till it converges then feed back the information to the MIMO detector. For better BER performance, it is possible that just performing a few iterations to get slightly better priors for the soft MIMO detector can improve the performance significantly. Such scheduling can also reduce the complexity/power of the iterative decoder significantly. One analytical method to create such an optimized scheduling can be the use of an EXIT (extrinsic information transfer) chart. By having, possibly, a three dimensional EXIT chart (between the iterative parts of the channel decoder, and the MIMO detector), a BER and power-optimal scheduling strategy will be explored.

B. Receiver algorithm development

After considering the average and worst case computational complexities of current state of the art receiver algorithms, we will focus on developing a new receiver structure. Current work that has been done on turbo MIMO receivers can be categorized as developing a new soft MIMO detector and pairing it with a channel code such as a convolutional code or turbo code. Soft detectors have been developed for the widely existing hard detectors, and a mix and match scheme has been used to implement each block of the fundamental architecture of iterative turbo MIMO receiver. Thus, by developing a new soft MIMO detection algorithm, a new class of iterative receiver can be created which can generate many different receivers based on the channel decoder type.

One example of such nascent detector is the Markov chain Monte Carlo (MCMC) [21] based MIMO detection. Computing the soft information based on all possible transmit symbols is a prohibitively expensive task. The main idea of the MCMC is to use Gibbs sampling to find a list of most likely pairs of transmitted sequence and then compute the soft information. The complexity of the MCMC will be dependent on the list size.

C. Architectural exploration of turbo MIMO receivers

Though there has been much work done on algorithmic exploration of iterative MIMO receivers, its implementation aspect has lacked attention. For one, the effect of implementation precision and quantization on the LLR calculation and overall power/performance tradeoff is yet to be explored.

Optimization of the iterative framework can also be explored. Much work has been done on the architecture of the components of an iterative MIMO receiver. Turbo decoders, turbo equalizers, MIMO detector architectures have all been studied as stand-alone blocks. Optimization of these existing blocks in the context of an iterative MIMO receiver will be beneficial as the work done on existing blocks can be utilized and leveraged. On the other hand, in the context of an iterative receiver, the implementation of the component blocks could differ considerably. Being aware of such soft information exchange could simplify the MIMO detector or channel decoder architecture. Once the iterative system and its components are optimized independently, it is also natural to consider joint optimization of the whole system.

In addition to the above, we could make use of the many common computational kernels between various soft MIMO detectors to develop an application specific instruction processor (ASIP) tailored for soft MIMO detection. Examples of such kernels include: matrix multiplication and QR decomposition in the preprocessing stage and LLR computation in the output stage. Commercially available ASIP design tools such as Target (www.retarget.com) or Coware (part of Synopsis) will be employed in this exploration.

REFERENCES

[1] P. Wolniansky, G. Foschini, G. Golden, and R. Valenzuela, "V-blast: an architecture for realizing very high data rates over the rich-scattering wireless channel," in *Signals, Systems, and Electronics, 1998. ISSSE 98. 1998 URSI International Symposium on*, Sep. 1998, pp. 295 –300.

[2] R. Kannan, "Improved algorithms for integer programming and related lattice problems," in *Proceedings of the fifteenth annual ACM symposium on Theory of computing*. ACM, 1983, pp. 193–206.

[3] J. Lagarias, H. Lenstra, and C. Schnorr, "Korkin-Zolotarev bases and successive minima of a lattice and its reciprocal lattice," *Combinatorica*, vol. 10, no. 4, pp. 333–348, 1990.

[4] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Mathematics of computation*, pp. 463–471, 1985.

[5] Z. Guo and P. Nilsson, "VLSI implementation issues of lattice decoders for MIMO systems," in *Proc. IEEE ISCAS*, May 2004, pp. IV – 477–480.

[6] W.-J. Choi, R. Negi, and J. Cioffi, "Combined ML and DFE decoding for the V-BLAST system," in *Communications, 2000. ICC 2000. 2000 IEEE International Conference on*, vol. 3, 2000, pp. 1243 –1248 vol.3.

[7] J. W. Choi, B. Shim, A. Singer, and N. I. Cho, "Low-complexity decoding via reduced dimension maximum-likelihood search," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1780–1793, Mar. 2010.

[8] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Transactions on Communications*, vol. 44, no. 10, pp. 1261 –1271, Oct. 1996.

[9] B. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Transactions on Communications*, vol. 51, no. 3, pp. 389 – 399, Mar. 2003.

[10] C. Studer, A. Burg, and H. Bolcskei, "Soft-output sphere decoding: algorithms and VLSI implementation," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 2, pp. 290 –300, Feb. 2008.

[11] M. Sellathurai and S. Haykin, "Turbo-BLAST for wireless communications: theory and experiments," *IEEE Transactions on Signal Processing*, vol. 50, no. 10, pp. 2538–2546, 2002.

[12] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *IEEE International conference on Communications (ICC)*, vol. 2. IEEE, 1993, pp. 1064–1070.

[13] C. Douillard, M. Jézéquel, C. Berrou *et al.*, "Iterative correction of intersymbol interference: Turbo-equalization," *European Transactions on Telecommunications*, vol. 6, no. 5, pp. 507–511, 1995.

[14] D. Zuyderhoff, X. Wautelet, A. Dejonghe, and L. Vandendorpe, "MMSE turbo receiver for space-frequency bit-interleaved coded OFDM," in *IEEE 58th Vehicular Technology Conference*, vol. 1. IEEE, 2003, pp. 567–571.

[15] J. Wang and S. Li, "MIMO turbo receiver with new probability data association soft interference cancellation," in *International Conference on Communications, Circuits and Systems*, vol. 1. IEEE, pp. 232–236.

[16] H. Vikalo, B. Hassibi, and T. Kailath, "Iterative decoding for MIMO channels via modified sphere decoding," *IEEE Transactions on Wireless Communications*, vol. 3, no. 6, pp. 2299–2311, 2004.

[17] Y. de Jong and T. Willink, "Iterative tree search detection for MIMO wireless systems," *IEEE Transactions on Communications*, vol. 53, no. 6, pp. 930–935, 2005.

[18] S. Chen and T. Zhang, "Low power soft-output signal detector design for wireless MIMO communication systems," in *Proc. IEEE ISLPED*, 2007, pp. 232–237.

[19] J. Ketonen, M. Juntti, and J. Cavallaro, "Performance-complexity comparison of receivers for a LTE MIMO–OFDM system," *IEEE Transactions on Signal Processing*, vol. 58, no. 6, pp. 3360–3372, 2010.

[20] "Catapult C synthesis datasheet," Mentor Graphics, Datasheet, 2008.

[21] B. Farhang-Boroujeny, H. Zhu, and Z. Shi, "Markov chain Monte Carlo algorithms for CDMA and MIMO communication systems," *IEEE Transactions on Signal Processing*, vol. 54, no. 5, pp. 1896–1909, 2006.