Implementation Aspects of Fixed-Complexity Soft-Output MIMO Detection

Di Wu, Erik G. Larsson and Dake Liu

Di Wu, Erik G. Larsson and Dake Liu, Implementation Aspects of Fixed-Complexity Soft-Output MIMO Detection, 2009, Proceedings of the 69th IEEE Vehicular Technology Conference (VTC'09), 1-5.
http://dx.doi.org/10.1109/VETECS.2009.5073587

Implementation Aspects of Fixed-Complexity Soft-Output MIMO Detection

Di Wu, Erik G. Larsson and Dake Liu
Department of Electrical Engineering, Linköping University, 581 83 Linköping, Sweden

***Abstract*—This paper discusses implementation aspects of a recently proposed fixed-complexity soft-output (FCSO) symbol detector for MIMO systems [4]. A further approximation to the FCSO detector is proposed which substantially reduces the complexity at the cost of a minor performance loss. With the resulting method, it is possible to carry out close-to ML detection for MIMO systems with a large number antennas (e.g. 4×4) using higher-order modulation schemes (e.g. 64-QAM) at low silicon cost in real-time. Furthermore, the parallelism inherited by the FCSO algorithm allows massive parallel processing which makes the method suitable for implementation in multi-core baseband signal processing hardware architectures.**

I. Introduction

Baseband signal processing in multiple-input multiple-output (MIMO) systems is a challenge because algorithms are usually several orders of magnitudes more complex than those used in single-input single-output (SISO) systems. While the scale of integration in semiconductors appears to continue follow Moore's law, the clock frequency of baseband hardware is limited by fundamental constraints such as power consumption and heat dissipation. In order supply more computing power, parallelism in the data processing is necessary.

One major problem in MIMO is signal separation/detection at the receiver. Essentially the problem is the following. Consider a MIMO system with n_t transmit antennas and n_r receive antennas. Let s be a transmitted vector of length n_t, obtained by mapping a set of information bits onto an M-QAM constellation $\mathcal{L}$. Then the received vector of length n_r is given by

$$y = Hs + n \tag{1}$$

where H is an $n_r \times n_t$ complex-valued channel matrix which is assumed to be known. The optimum soft detector should compute

$$L(b_i|y) = \log\left(\frac{\sum_{s:b_i(s)=1}\exp(-\frac{1}{N_o}\|y-Hs\|^2)}{\sum_{s:b_i(s)=0}\exp(-\frac{1}{N_o}\|y-Hs\|^2)}\right) \tag{2}$$

Here "$s : b_i(s) = \beta$" means all s for which the ith bit of s is equal to β. Computing (2) requires enumeration of the entire set of possible transmitted vectors. The complexity of doing this is usually not affordable for implementation in practice.

The work of D. Wu and D. Liu was supported in part by the European Commission through the EU-FP7 Multi-base project and the Swedish Foundation for Strategic Research (SSF). The work of E. Larsson was supported in part by the Swedish Research Council (VR), the Swedish Foundation for Strategic Research (SSF) and VINNOVA. E. Larsson is a Royal Swedish Academy of Sciences (KVA) Research Fellow supported by a grant from the Knut and Alice Wallenberg Foundation.

Commonly the sums in (2) are approximated by their largest terms. This approximation is called "log-max". This requires solving problems of the type $\min\|y - Hs\|$, subject to a finite-alphabet constraint on s. This can be solved with good performance for example, via sphere decoding [1], [2]. The complexity of sphere decoding, however, grows exponentially with the number of transmit antennas and polynomially in the size of the signal constellation. More importantly, the tree search used in sphere decoding is in principle a sequential procedure which is difficult to parallelize. In [3], a fixed-throughput sphere detector was proposed. The idea of [3] was to partition the transmitted symbols into different layers (ordered according to the SNR per symbol), and then apply a full ML search to the top (most uncertain) layers and a linear equalizer to the rest of the layers. This algorithm has the advantage of fixed and relatively low complexity. However, it only provides hard decisions and the log-max philosophy must be used to approximate (2).

Soft-output approximations to (2) which are not based on log-max also exist. We will here study the fixed-complexity soft-output (FCSO) MIMO detector proposed in [4]. This detector replaces the likelihood marginalization ($\sum$ in (2)) by an exact summation over a subset of the transmitted bits and an approximate maximization over the rest. The bits over which exact marginalization is performed are the most uncertain ones, that is the ones that are received with lower signal-to-noise-ratio (SNR). The resulting approximation of (2) is shown in (3), where marginalization is assumed to take place over r entire complex symbols (rather than bits), an assumption we make hereafter. (In this equation, the symbols are assumed to be sorted in the optimal order. Also, $\mathscr{H}$ is the channel matrix with columns $1, ..., r$ removed.) It was shown in [4] that the approximate marginalization step (the max operation in (3)) can be carried out using a suboptimal method (ZF-DFE was suggested in [4]) with little loss of performance. The FCSO method has low complexity and a fully parallel structure which makes it suitable for implementation in parallel hardware. Additionally, its complexity is fixed, and this makes it suitable for pipelined implementation.

In this paper:

- A reduced-complexity variant of the FCSO MIMO detector [4] for high-order modulation schemes is proposed. In effect, the proposed detector is a further approximation of that in [4]. The approximation consists of only partially enumerating the symbols selected for exact marginalization (the set $\mathcal{L}$ in (2)).
- Several QR-decomposition (QRD) algorithms are com-

$$L(b_i|y) \approx \log\left(\frac{\sum_{s_1\in\mathcal{L}}\cdots\sum_{s_r\in\mathcal{L}} \max_{s_{r+1},\ldots,s_{n_t}} \exp(-\frac{1}{N_o}\|y - h_1s_1 - \ldots - h_rs_r - \mathscr{H}(s_{r+1},\ldots,s_{n_t})^T\|^2)}{\sum_{s_1\in\mathcal{L}}\cdots\sum_{s_r\in\mathcal{L}} \max_{s_{r+1},\ldots,s_{n_t}} \exp(-\frac{1}{N_o}\|y - h_1s_1 - \ldots - h_rs_r - \mathscr{H}(s_{r+1},\ldots,s_{n_t})^T\|^2)}\right) \quad (3)$$

pared to further reduce the complexity of the preprocessing done once per channel realization while supplying sufficient numerical stability to the MIMO detection.
- The complexity of the method of [4] and that of the proposed method is analyzed in detail for a 4×4 MIMO system with 64-QAM modulation in terms of real-valued arithmetic operations, in order to assess the amount of hardware resources needed.

II. Fixed-Complexity MIMO Detection

A. Fixed-Complexity Soft-Output (FCSO)

We take a 4×4 MIMO system using 64-QAM as a case study. In this paper, each complex-valued symbol is considered to be one layer and only the top layer is exactly marginalized with the remaining three layers approximately marginalized. The channel-rate processing of FCSO involves the QRD of n_t rank-reduced channel matrices

$$\mathscr{H}_{\bar{k}} = [h_1, ..., h_{k-1}, h_{k+1}, ..., h_{n_t}] \quad (4)$$

which generates an upper triangular matrix $R_{\bar{k}}$, and a unitary matrix $Q_{\bar{k}}$ so that

$$\mathscr{H}_{\bar{k}} = Q_{\bar{k}}R_{\bar{k}} \quad (5)$$

Here n_t QRD is needed for different $\mathscr{H}_{\bar{k}}$.

The symbol-rate processing consists of the following steps:

1) Pick one transmitted symbol $s_i, i \in (1,\ldots,n_t)$ as the top layer. The entire constellation $\mathcal{L}$ is enumerated in the exact marginalization ($\sum$ in (3)) only for s_i. For the k^{th} candidate $\hat{s}_i^k$ in $\mathcal{L}$, by canceling its effect on the received symbol vector y, a new vector

$$\hat{y} = y - \tilde{h}_i\hat{s}_i^k \quad (6)$$

is computed.

2) By multiplying $\hat{y}$ with $Q_{\bar{k}}^H$ from (5), compute

$$\tilde{y} = Q_{\bar{k}}^H\hat{y} \quad (7)$$

3) Based on $\tilde{y}$ and R, using DFE, $\hat{s}_b = [\hat{s}_2\hat{s}_3\hat{s}_4]^T$ can be estimated using hard decisions. From this, compute the Euclidean distance

$$\delta_k = \|\hat{y} - R_{\bar{k}}\hat{s}_b\|^2 \quad (8)$$

and eventually the log-likelihood ratio (LLR)

$$\mu(b_1,\cdots,b_{24}) = \exp\big(-\frac{1}{N_o}\delta_k\big) \quad (9)$$

The LLR of the six bits that constitute the top-layer symbol can be computed using (8). This involves the computation of 64 different $\delta_k, (k = 1,\cdots,64)$

B. Layered Orthogonal Lattice Detector (LORD)

The Layered Orthogonal Lattice Detector (LORD) is proposed in [5]. It is similar to FCSO with only one layer used in the exact marginalization (that is the case considered here). In [5], the system is represented through a lattice formulation:

$$S = [s_{1,I}, s_{1,Q}, ..., s_{n_t,I}, s_{n_t,Q}]^T \quad (11)$$

Instead of finding the maximum value of the metric over all possible values of s_{n_t} (3), the search can be greatly simplified by noting for given values of $s_{n_t,I}$ and $s_{n_t,Q}$ the maximum-likelihood (ML) metric can be approximated using DFE. This implies that the number of points that has to be searched in this formulation is M (with two slicing operations ($s_{n_t,I}$ and $s_{n_t,Q}$) per searched point). For example, for 64-QAM MIMO systems ($M = 64$), every symbol pair in S will be taken as the top layer pair once to compute the approximated ML metrics. n_t cyclic shifts are then applied so that every real-valued symbol pair ($s_{i,I}$ and $s_{i,Q}$) will eventually be shifted to the last position in the vector S. This implies that the DFE approximation needs to be computed $4\times 64 = 256$ times. A difference from [4] is that, instead of using (2), the max-log approximation in (12) is used to compute the LLR values:

$$L(b_i|y) = \log\left(\frac{\max_{s:b_i(s)=1}\exp(-\frac{1}{N_o}\|y - Hs\|^2)}{\max_{s:b_i(s)=0}\exp(-\frac{1}{N_o}\|y - Hs\|^2)}\right) \quad (12)$$

This is equivalent to

$$L(b_i|y) = -\frac{1}{N_o}(\min_{s:b_i(s)=1}\|y - Hs\|^2 - \min_{s:b_i(s)=0}\|y - Hs\|^2) \quad (13)$$

Essentially, [4] is more general than [5] in that it allows more than one layer (either real or complex symbols) to be exactly marginalized. Nevertheless, for MIMO with no more than four antennas, fully enumerating one layer complex symbol already achieves close-to-ML performance with a reasonable complexity.

III. Modified FCSO (MFCSO)

Although the FCSO MIMO detector presented in [4] already has very low complexity compared to an ML detector, further reduction is desired for a practical implementation with large signal constellations. In the following, further approximations and improvements to FCSO detection [4] will be proposed, and their complexity will be studied.

In [4], the entire constellation $\mathcal{L}$ is enumerated in the exact marginalization ($\sum$ in (3)). In this paper, instead of searching the full constellation $\mathcal{L}$, we propose to sum over only a subset $\mathcal{L}_s \subset \mathcal{L}$ of constellation points around an initial estimate $\hat{s}$. This initial estimate will be obtained by zero-forcing detection. The size of $\mathcal{L}_s$, denoted by N, is chosen to be 16 and 8 in this paper for the complexity and performance comparisons. In effect, the proposed detector is a further approximation of that in [4], it consists of only partially enumerating the symbols selected for exact marginalization (the set $\mathcal{L}$ in (3)).

Similarly to FCSO, the channel-rate processing of MFCSO involves computing a QRD n_t times, as shown in (5) and (4).

$$L(b_i|y) \approx log\left(\frac{\sum_{b(\hat{s}_1)_1=0}^{1}\cdots\sum_{b(\hat{s}_1)_{i-1}=0}^{1}\sum_{b(\hat{s}_1)_{i+1}=0}^{1}\cdots\sum_{b(\hat{s}_1)_6=0}^{1}\left(\max_{b(\hat{s}_2)_1,\cdots,b(\hat{s}_4)_6}\mu(b_1,\cdots,b_{i-1},1,b_{i+1},...,b_{24})\right)}{\sum_{b(\hat{s}_1)_1=0}^{1}\cdots\sum_{b(\hat{s}_1)_{i-1}=0}^{1}\sum_{b(\hat{s}_1)_{i+1}=0}^{1}\cdots\sum_{b(\hat{s}_1)_6=0}^{1}\left(\max_{b(\hat{s}_2)_1,\cdots,b(\hat{s}_4)_6}\mu(b_1,\cdots,b_{i-1},0,b_{i+1},...,b_{24})\right)}\right) \quad (10)$$

As an overhead compared to FCSO, the coefficient matrix

$$W = (H^H H + \sigma^2 I)^{-1} H^H \quad (14)$$

is needed to perform the ZF/MMSE based initial estimate of $\hat{s}$ in (15) below.

The symbol-rate processing of MFCSO is the following:

1) Linear detection (ZF/MMSE) is carried out to estimate the initial symbol vector

$$\hat{s} = \min_{s_k \in \mathcal{L}} \|Hs - y\|^2 \quad (15)$$

Here s is the transmitted symbol vector, s_k is the k^{th} symbol in it.

2) For each initially estimated symbol $\hat{s}_k, k \in \{1, ..., n_t\}$, a candidate set $\mathcal{L}_k$ is created. $\mathcal{L}_k$ contains N lattice points close to $\hat{s}_k$.

3) For each point $l \in \mathcal{L}_k$, approximate marginalization is applied to the rest of the layers either via ZF or ZF-DFE. According to (16), a multiplication of $Q_{\bar{k}}^H$ and $\hat{y}$ is needed for each $\hat{y}$ which is updated proportionally to the size of $\mathcal{L}_k$ and the symbol rate. However, note that

$$\widetilde{y} = Q_{\bar{k}}^H \hat{y} = Q_{\bar{k}}^H (y - h_k l) = Q_{\bar{k}}^H y - (Q_{\bar{k}}^H h_k) l \quad (16)$$

where $Q_{\bar{k}}^H h_k$ is an $n_t \times 1$ vector, which can be precalculated at channel rate.

4) Using back-substitution [7], $\hat{s}_b$ can be estimated from

$$\hat{s}_b = \arg\min_{s_k \in \mathcal{L}} \|R_{\bar{k}} \hat{s}_b - \widetilde{y}\|^2 \quad (17)$$

5) $\hat{s}_b$ together with $\hat{s}_k$ form a complete possible transmitted symbol vector which has an Euclidean distance

$$\delta_l = \|R_{\bar{k}} \hat{s}_b - \widetilde{y}\|^2 \quad (18)$$

6) In total, there will be N different $l \in \mathcal{L}$ values for each layer, and there will be four layers each being the top layer once. Therefore, for the 4×4 system, $4N$ different δ_l values need to be computed. In case $N = 16$, there will be 64 different δ_l values which is 1/4 compared to the FCSO proposed in [4].

7) For the sake of low complexity, instead of MAP detection, an approximation can be adopted to compute $L(b_i(s_k))$ as explained in Sec. V-C.

IV. Simulations

In order to evaluate the performance of the MFCSO method proposed in this paper, a C++ model of a 4×4 MIMO system using 64-QAM was developed based on the IT++ libraries. Both slow and fast fading channels and two different convolutional coding (CC) rates (1/2, 1/3 and 3/4) were used in the simulation. Monte-Carlo simulation was used to compute the FER/SNR curves. By default, FER= 10^{-2} is considered to be acceptable in practice for data communications.

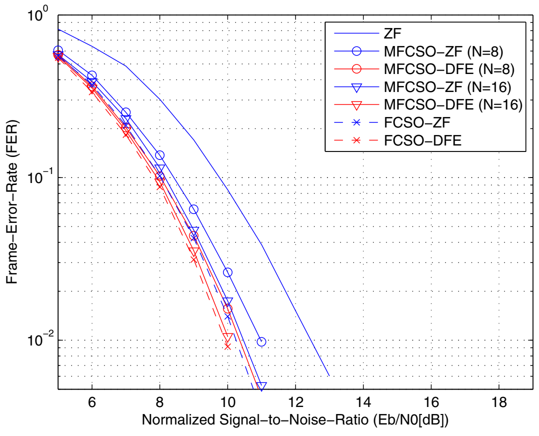

Fig. 1: 4×4 64-QAM system (fast fading, rate-1/2 CC)

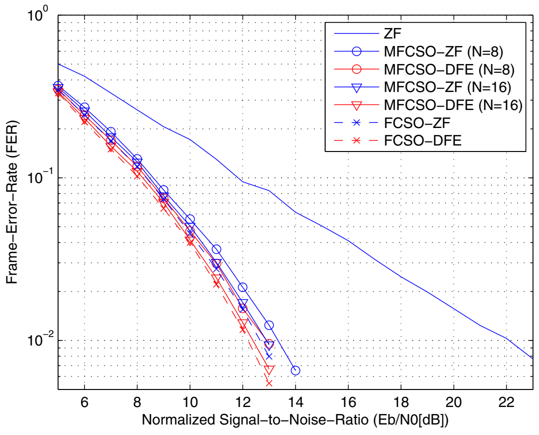

Fig. 2: 4×4 64-QAM system (slow fading, rate-1/3 CC)

In Fig. 1 and Fig. 2, the packet length is 100 (data bits) and the coderates are 1/2 and 1/3, respectively. Fig. 1 shows that in fast-fading scenarios, the linear detector can exploit the time diversity; hence the ZF error rate curve has a similar shape as those for MFCSO. Fig. 2 shows that in slow fading scenarios, the performance gap between MFCSO and ZF is significant. More importantly, it also shows that the performance loss brought by the approximation ($N = 16$ and $N = 8$) proposed in this paper is very small, and that MFCSO-DFE has a small gain over the MFCSO-ZF ($\sim$ 0.25dB).

The simulations in Fig. 3 use a packet length of 1000 and a convolutional code with rate 3/4. The result shows that MFCSO-DFE with $N = 16$ has about 0.3 dB degradation compared to $N = 64$. An 1.2 dB degradation is incurred when $N = 8$. Therefore, $N = 16$ may be chosen as a trade-off for realistic MIMO systems.

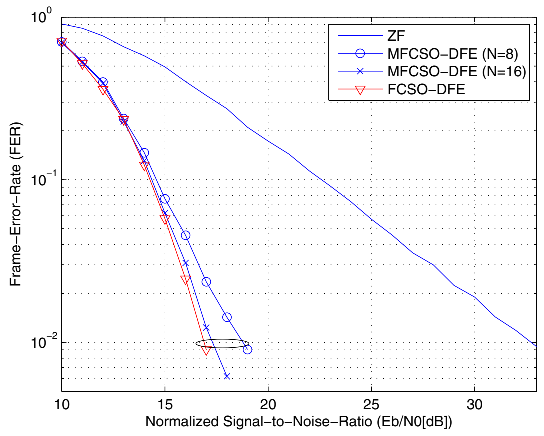

Fig. 3: MFCSO with different N (slow fading, rate-3/4 CC)

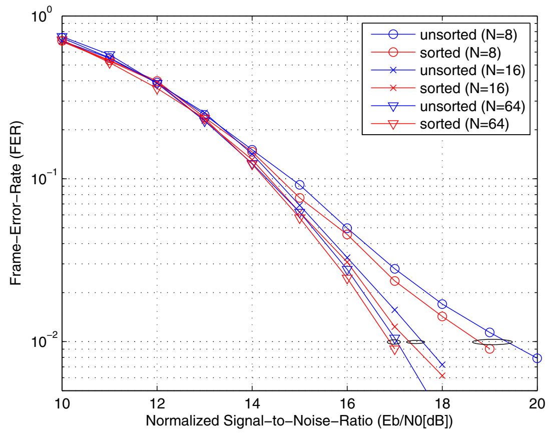

Fig. 4: Sorted vs. unsorted (slow-fading, rate-3/4 CC)

V. Implementation Aspects

A. Sorted vs. Unsorted ZF-DFE for Approximate Marginalization

The optimal symbol ordering for DFE comes at a complexity cost which is not negligible in practice. In this paper, a heuristic sorting is applied during the DFE approximation procedure. The index vector is found according to

$$\beta = \text{sort}(\text{diag}((\mathscr{H}_{\bar{k}}^H \mathscr{H}_{\bar{k}})^{-1})) \qquad (19)$$

where β contains the indices of the sorted diagonal values. For example, after sorting $\mathscr{H}_{\bar{1}}$, a new matrix

$$\tilde{\mathscr{H}}_{\bar{1}} = [\tilde{h}_2 \tilde{h}_3 \tilde{h}_4] \qquad (20)$$

which is a column-reordered $\mathscr{H}_{\bar{k}}$ in (4) is generated, where

$$\tilde{h}_i = h_{\beta(i)}, i \in \{1, 2, 3\} \qquad (21)$$

is one column in the new channel matrix. The sorting involves a few extra operations in the channel rate processing. Fig. 4 (same simulation setup as in Figure 3) shows that the gain brought by sorting is less than 0.5 dB compared to MFCSO-DFE without sorting. This is due to the small size of $\mathscr{H}_{\bar{k}}$. The gain is expected to grow as the size of H increases.

Method	**Brute-force GR**	**MGS with QR Updating**	**MGS**
Mul	1948	900	1164
Add/Sub	1000	592	1180
$\frac{1}{x}/\frac{1}{\sqrt{x}}$	24	10	12

TABLE I: QRD complexity of four $\mathscr{H}_{\bar{k}}$ in (4) (no. of real ops)

B. QR decomposition (QRD) Strategy

Since multiple QRDs are involved in the channel-rate processing of FCSO, it is a significant burden which requires simplification. As mentioned in [7], there exist different ways of computing the QRD. Among these methods, Givens Rotation (GR) with QR updating and Modified Gram-Schmidt (MGS) are investigated in this paper as candidates for QRD.

1) GR-based QR Updating: In order to compute the LLR of the first symbol, a QRD of $\mathscr{H}_{\bar{1}}$ is needed. This matrix is a thin matrix obtained by deleting the first column of H. Instead of computing the QRD of $\mathscr{H}_{\bar{1}}$ from scratch, it can be computed based on Q and R decomposed from the original H which is a square matrix. By deleting the first column of R, a new matrix

$$\hat{R} = \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \end{bmatrix}$$

is generated. In order to convert $\hat{R}$ into an upper-triangular matrix, three Givens rotations can be applied to zero-out the three lower-off-diagonal values. The basic idea of GR [7] is to introduce zeros by rotating rows in the target matrix A with a series of rotation matrices G_i so as to convert it into upper triangular form (R) while accumulating the rotations into Q so that $QR = A$. Each rotation involves two rows and it annihilates (zeros out) one designated element in one of the rows:

$$R_{\bar{1}} = G_3 G_2 G_1 \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \end{bmatrix} = \begin{bmatrix} \times' & \times' & \times' \\ 0 & \times' & \times' \\ 0 & 0 & \times' \\ 0 & 0 & 0 \end{bmatrix}$$

where G_i are rotation matrices. For example, to zero-out $\hat{R}(2,1)$, G_1 is applied to row 1 and 2. In a similar manner, the QRD of $\mathscr{H}_{\bar{2}}$ can be generated by two GR, and only one Givens rotation is needed to compute that of $\mathscr{H}_{\bar{3}}$. The matrix $R_{\bar{4}}$ is just R with the last column removed and $Q_{\bar{4}} = Q$. In total, six GR are needed to compute four pairs of $Q_{\bar{k}}$ and $R_{\bar{k}}$. When compared to four brute-force GR based QRD, around half operations can be saved.

2) MGS: The MGS method generates an orthonormal vector set in an inner product space. For an $m \times n$ matrix A = $[a_1 \ldots a_n]$, to compute the i^{th} vector q_i in the unitary matrix $Q = [q_1 \ldots q_n]$, Although MGS cannot reuse values computed earlier as compared to GR with QR updating, it involves fewer operations than GR method. Note $\mathscr{H}_{\bar{k}}$ in (16) is a 4×3 tall matrix, hence $Q_{\bar{k}}$ is not a unitary matrix. When computing δ_k, a penalty term $\|(I - Q_{\bar{k}} Q_{\bar{k}}^H)\hat{y}\|^2$ needs to be added so that $\delta_l = \|\hat{y} - \mathscr{H}_{\bar{k}} s\|$. This requires precomputation of $I - Q_{\bar{k}} Q_{\bar{k}}^H$ and $(I - Q_{\bar{k}} Q_{\bar{k}}^H) h_k$ as a part of the channel-rate processing and also extra computations in the symbol-rate processing.

The complexities of the various QRD strategies are shown in Tab. I. This table shows that when using MGS to compute the initial Q and R and then applying QR updating (namely MGS with QR Updating in Tab. I), the fewest operations are

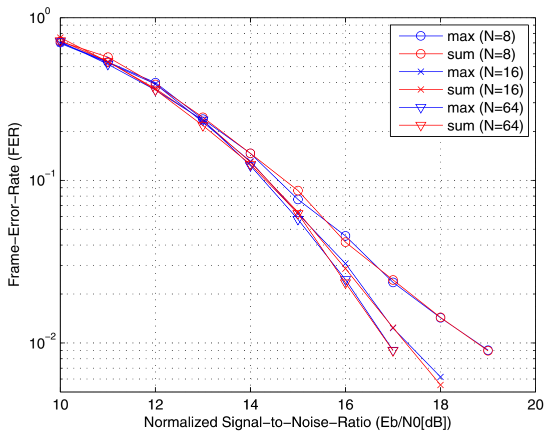

Fig. 5: Max vs. sum (slow fading, rate-3/4 CC)

needed to compute four $\mathscr{H}_k$. Hence it is used in this paper, and the complexity figures in Tab. II and III (see Section VI) are calculated based on this method.

C. Summation versus Maximum

The sum in the numerator and denominator of (2) can be computed recursively, exploiting the so-called Jacobian logarithm:

$$\log(e^a + e^b) = \max(a, b) + \log(1 + e^{-|a-b|}) \qquad (22)$$

which can be implemented efficiently and with good numerical stability, even in fixed-point arithmetic. In particular, the second term can be implemented via a table-lookup as a function of $|a - b|$ [4]. By omitting the second term, a further approximation (namely log-max) can be obtained, resulting in the following approximation for $L(b_i|y)$:

$$L(b_i(s_k)) \approx -\frac{1}{\sigma^2}\{\min_{\mathbf{l}\in\mathcal{L}_\mathbf{k}:\mathbf{b_i(s_k)=0}} \delta_\mathbf{l} - \min_{\mathbf{l}\in\mathcal{L}_\mathbf{k}:\mathbf{b_i(s_k)=1}} \delta_\mathbf{l}\} \quad (23)$$

A performance comparison of MFCSO using (22) and (23) is shown in Fig. 5.

VI. Complexity Assessment

Reference [4] provided only asymptotic order-type complexity arguments, and no detailed information was given on the implementation complexity. In this paper, in order to make a quantitative comparison, the complexity is broken down into real-valued arithmetic operations for the case of 4×4 MIMO with 64-QAM, using exact marginalization over one layer of complex symbols ($r = 1$ in (3)). The complexities of FCSO and MFCSO are compared in the following.

The complexity counts can be broken down into two parts: channel-rate processing (things which are done once per channel coherence interval [for every new channel matrix H], e.g. QR decomposition or matrix inversion) and symbol-rate processing (things that are done for each vector y, e.g., visiting nodes in a tree-search algorithm approximation of (3)). Table II shows the computational complexity of FCSO, for $r = 1$ [complex symbol] and using the approximations and improvements introduced above.

Method		FCSO-ZF	FCSO-DFE	ZF
Channel-rate processing (T_c)	Mul	1404	1404	550
	Add/Sub	1114	1114	536
	$\frac{1}{x}/\frac{1}{\sqrt{x}}$	22	22	2
Symbol-rate processing (T_s)	Mul	15616	15640	576
	Add/Sub	13824	14080	832

TABLE II: Complexity of FCSO (number of real ops)

Method		MFCSO-ZF		MFCSO-DFE	
Channel-rate Preprocessing	Mul	1754		1754	
	Add/Sub	1388		1388	
	$\frac{1}{x}/\frac{1}{\sqrt{x}}$	24		24	
N (Size of $\mathcal{L}$)		16	8	16	8
Symbol-rate processing (T_s)	Mul	4160	2240	4184	2264
	Add/Sub	3712	2016	3776	2048

TABLE III: Complexity of MFCSO (number of real ops)

The complexity of MFCSO is given in Table III. Note that the channel-rate processing complexity of MFCSO is slightly higher than that of FCSO because of the extra processing (15) needed for the initial estimation. The result shows that the MFCSO method proposed has a symbol-rate complexity which is only 1/4 (for $N = 16$) and 1/7 (for $N = 8$) compared to the FCSO [4].

VII. Conclusion

Our results indicate that when using a strong channel code (e.g. 1/2 CC), MFCSO-DFE method with $N = 8$ achieves a good performance/complexity tradeoff with a symbol-rate complexity reduction in the order of $N = 8$ compared to the original FCSO (where $N = 64$ for $\mathcal{L}$). For a MIMO system with weaker coding (e.g. 3/4 CC), $N = 16$ is preferred as depicted in Fig. 3. Even with $N = 16$, the symbol-rate complexity is only 1/4 of FCSO. This is a substantial reduction considering the tough constraints on silicon cost and power consumption for baseband chips. In summary the results indicate that MFCSO is a promising close-to-ML detection approach.

References

[1] E. Viterbo and J. Boutros, "A universal lattice code decoder for fading channels", *IEEE Trans. Inform. Theory*, vol. 45, no. 5, pp. 1639-1642, July 1999.

[2] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bölcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm", *IEEE Journal of Solid-State Circuits*, vol. 40, no. 7, pp. 1566-1577, July 2005.

[3] L. G. Barbero, J. S. Thompson, "Rapid prototyping of a fixed-throughput sphere decoder for MIMO systems", in *Proc. of IEEE ICC*, Jun. 2006.

[4] E. G. Larsson and J. Jaldén, "Soft MIMO detection at fixed complexity via partial marginalization", *IEEE Transactions on Signal Processing*, vol. 56, pp. 3397-3407, Aug. 2008.

[5] M. Siti and M. P. Fitz, "A novel soft-output layered orthogonal lattice detector for multiple antenna communications", in *Proc. of IEEE ICC*, Jun. 2006.

[6] P. Fertl, J. Jaldén and G. Matz, "Capacity-based performance comparison of MIMO-BICM demodulators,", in *Proc. of IEEE SPAWC*, 2008.

[7] G. H. Golub, C. F. Van Loan, *Matrix Computations*, Third Edition, The Johns Hopkins University Press, 1996.