Allocation of Computational Resources for Soft MIMO Detection

Mirsad Čirkić, Daniel Persson and Erik G. Larsson

Mirsad Čirkić, Daniel Persson and Erik G. Larsson, Allocation of Computational Resources for Soft MIMO Detection, 2011, accepted IEEE Journal of Selected Topics in Signal Processing.

Postprint available at: Linköping University Electronic Press
http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-69612

Allocation of Computational Resources for Soft MIMO Detection

Mirsad Čirkić, Daniel Persson, and Erik G. Larsson
Communication Systems Division, Department of Electrical Engineering (ISY)
Linköping University, SE-581 83 Linköping, Sweden. {mirsad,danielp,erik.larsson}@isy.liu.se

***Abstract*—We consider soft MIMO detection for the case of block fading. That is, the transmitted codeword spans over several independent channel realizations and several instances of the detection problem must be solved for each such realization. We develop methods that adaptively allocate computational resources to the detection problems of each channel realization, under a total per-codeword complexity constraint. Our main results are a formulation of the problem as a mathematical optimization problem with a well-defined objective function and constraints, and algorithms that solve this optimization problem efficiently computationally.**

I. Introduction

We consider the problem of optimizing the computational resources allocated for soft detection of coded bits in a single-user multiple-input multiple-output (MIMO) system. It is well known that MIMO systems can substantially increase the capacity by exploiting spatial diversity in rich scattering environments [1], [2]. One key component of a MIMO system is the signal separation/detection at the receiver. This task is difficult and computationally expensive, especially when the wireless environment yields "bad", i.e., ill-conditioned MIMO channels. For instance, the complexity of the soft maximum likelihood (ML) method, which solves these detection problems optimally, grows exponentially with the number of antennas and polynomially with the size of the signal constellation. Sub-optimal linear methods, such as soft zero-forcing (ZF), which have much lower complexity, are however often sufficiently accurate for "good" channels. In such a case, performing ML detection would be a waste of computational power. This said, for MIMO detection there is always a trade-off between performance and complexity. The fundamental question is, can we save computational resources by performing ML detection only when it is needed, and something simpler when it is not?

There is a variety of methods available that approximate the ML detector, see e.g. [3] and the references therein. The more advanced methods offer some means of trade-off between computational complexity and performance by adjusting a parameter, e.g., max-log with the sphere decoder (SD) [4], max-log with the fixed complexity SD (FCSD) [5], and soft-output via partial marginalization (PM) [6]. The PM method is attractive as it provides a simple and well-defined way of trading complexity for performance, it is highly parallelizable, and has an easily predictable fixed run-time. Additionally, it has recently been shown that when a priori information is incorporated, the PM method can outperform the pure max-log method, and inherently all max-log based methods, see [7]. Therefore, we exemplify our core ideas with the PM method. Nevertheless, the basic ideas and methods that we propose in this paper can be used with any MIMO detector of choice.

In practice, the information is typically spread out by coding over both frequency and time, and the channel may change with both frequency and time as well. As to what concerns the channel fading, and specifically how much the MIMO channel varies within a codeword, there are two extreme cases. The first case is when the channel does not change within a codeword, so that each transmitted symbol vector sees the same realization of the channel. The allocation of computational resources becomes trivial in this case since it is then natural to use the same detector for all symbol vectors. The other extreme case is when the channel changes independently from each symbol vector to the next. Then, simple, linear detectors often work sufficiently well, see [1, p. 362]. Against this background, we would expect that the potential gains of computational resource allocation are largest when the number of channel realizations that each codeword spans over is modest, say from 2 up to in the order of, say, 50 realizations per codeword. The operating regime where our technique has the largest potential in practice depends both on the coherence time and coherence bandwidth of the channel, and on the specific coding scheme in use. For example, suppose that codewords are obtained by coding over the frequency domain, as is typical of OFDM systems. Then, for our technique to deliver an appreciable gain the channel must offer some frequency diversity, but not "too much". The amount of frequency diversity offered by the channel is dictated by its coherence bandwidth. To give some numbers, note that in [8], the ITU Vehicular A channel model has a coherence bandwidth of about 0.25 MHz and the Vehicular B model has a coherence bandwidth of 2.7 MHz. This means that with a 5 MHz allocated bandwidth and no mobility (or no coding over time), the number of channel realization per codeword would be in the order of 20 and 2, for models A and B, respectively. Hence, it is plausible that there are practical scenarios where computational resource allocation can offer actual performance gains.

The total computational power that can be spent on de-

This work was supported in part by the Swedish Research Council (VR), the Swedish Foundation of Strategic Research (SSF), and the Excellence Center at Linköping-Lund in Info. Tech. (ELLIIT). E. G. Larsson is a Royal Swedish Academy of Sciences (KVA) Research Fellow supported by a grant from the Knut and Alice Wallenberg Foundation.

tection is limited and it is therefore important to utilize it optimally. In general, we want to assign a performance-complexity tradeoff parameter to each detection problem to be solved, and optimize this parameter based on the properties of the corresponding channel matrix and on the amount of available computational resources for the codeword. We refer to this general case as the "multi-decision" scenario. An illustrative special case is when we can only make a binary decision on whether or not to detect a specific received bit. In such a "binary-decision" scenario, we can exploit the fact that in a coded system it is not necessary to correctly detect all the bits of a codeword, but that it suffices to detect a portion of the codeword and treat the rest of the bits as erasures. Given a predetermined maximal allowed per-codeword complexity, which part of the codeword should we detect and which part should we skip?

To our knowledge, there is not much earlier work that considers detection with adaptive allocation of computational resources. Some general ideas were outlined in [9] and some more specific aspects of the problem were addressed in [10], [11], and [12]. The work of [10] is specific to the SD where a maximum allowed per-codeword as well as a per-symbol time-limit (complexity) is set and the symbols are detected on a first-in first-detected basis. The first symbols that fit within the predetermined time-limits are detected, and the symbols that are left over are ignored. This setting is similar to the "binary-decision" scenario described in the previous paragraph, but without performing any ordering. As we will see in this work, proper ordering can significantly increase the performance. In [11], an SNR-asymptotic BER approximation is derived for the ZF and ML detectors, which is then used to predict whether ZF or ML should be used on different symbols without violating a predetermined minimum BER. This approach is also binary in the sense that either ZF or ML is performed, but nothing in between. By contrast, our approach, used together with the PM MIMO detector, allows us to perform this adaptive detection in a "multi-decision" fashion. In [12], the ideas of [10] are extended by setting a minimum accuracy threshold (smallest absolute value) for each evaluated soft bit. For some bits, it is computationally more expensive to meet the accuracy threshold than for others. This threshold is derived in order to assure a certain performance.

The main contribution of this paper is a general framework for adaptive allocation of computational resources. Specifically, we formulate the allocation problem as an optimization problem with a well-defined objective function and constraints. We devise optimal and sub-optimal algorithms that solve this optimization problem in a computationally efficient manner. The framework can handle problems both of the binary-decision and the multi-decision type. The work in this paper is a comprehensive extension of our conference papers [13], [14].

II. Preliminaries

A. Model

We use a real-valued block fading MIMO-channel

$$\boldsymbol{y}_{k,\ell} = \boldsymbol{H}_k \boldsymbol{s}_{k,\ell} + \boldsymbol{e}_{k,\ell}, \tag{1}$$

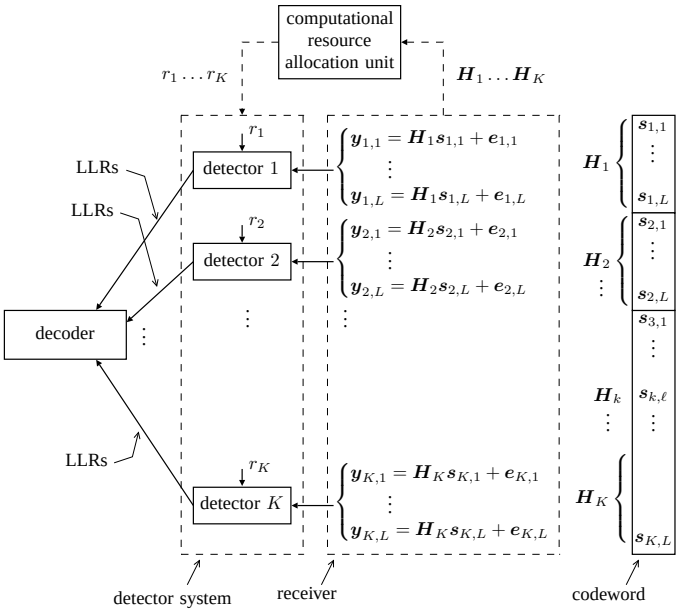


Fig. 1: Proposed receiver structure. The figure illustrates different components of the communication chain at the receiver side. The soft-output (LLR) is defined in (2) and the parameter r_k decides the trade-off between complexity and performance for a channel matrix $\boldsymbol{H}_k$. The indices $k = \{1, \dots, K\}$ and $\ell = \{1, \dots, L\}$ apply for a codeword that contains KL symbol vectors with N_{T} symbols per vector and that has propagated through the channels $\boldsymbol{H}_1, \dots, \boldsymbol{H}_K$.

where for one encoded data packet, i.e., codeword, we have K independent channel realizations $\boldsymbol{H}_k \in \mathbb{R}^{N_{\mathrm{R}} \times N_{\mathrm{T}}}$, and where for each k we transmit $L > 1$ symbol vectors $\boldsymbol{s}_{k,\ell} \in \mathcal{S}^{N_{\mathrm{T}}}$ where $\ell \in \{1, \dots, L\}$, see Fig. 1. Further, $\boldsymbol{e}_{k,\ell} \in \mathbb{R}^{N_{\mathrm{R}}} \sim \mathcal{N}(\boldsymbol{0}, \frac{N_0}{2}\boldsymbol{I})$ denotes the noise vector and the total per-codeword number of received vectors $\boldsymbol{y}_{k,\ell} \in \mathbb{R}^{N_{\mathrm{R}}}$ is KL. For simplicity, we use a BPSK constellation $\mathcal{S} = \{-1, +1\}$. However, the methods that we present in this paper can be extended to other constellations as well. The channel is perfectly known to the receiver and in what follows, $N_{\mathrm{R}} \geq N_{\mathrm{T}}$ since it is typical in practice and simplifies the mathematics performed in this paper. Every complex-valued model of the type (1) can be posed as a real-valued model of the same type if the complex symbol constellation is separable, see [6].

B. Setup

The proposed receiver structure for our problem setup is presented in Fig. 1. A data packet is transmitted and recovered at the receiving side. This data packet is first encoded by a bit-interleaved channel encoder, such as a low-density parity-check (LDPC) encoder for instance, which adds parity bits and spreads the bits over the codeword. This setting is referred to as bit-interleaved coded modulation (BICM), which is common in modern communication systems, see [15]. After modulation, the coded symbols are transmitted through the wireless channel, which we model by (1). Since we are looking at a block fading system, for the k:th channel realization, we have a group of L detection problems to solve. Each group k will differ in performance depending on how "good" the corresponding k:th channel realization is. The computational resource allocation unit in Fig. 1 distributes the computational

resources appropriately amongst these K channel realizations. This unit presumes that a parameter r_k exists for each detector in Fig. 1 by which complexity can be traded for performance. For example, detectors such as PM and SD facilitate this tradeoff via the r-parameter and the sphere radius parameter, respectively.

Fig. 1 represents a receiver architecture with the purpose of illustrating the main ideas of this paper. Provided that K is not too small, the number of detectors to which the computational resource allocation unit assigns a given complexity parameter r_k would be approximately the same for each codeword. Hence, a more practical implementation may comprise a number of detectors designed for fixed r_k, and a small number of reconfigurable detectors that can handle a varying r_k. Then, the computational allocation unit would redirect the received vectors to the appropriate detector, rather than to supply each single detector with a new value of r_k.

C. Soft Detection

For the sake of simplicity, we omit the indices k and ℓ in the remaining parts of this paper, unless explicitly stated otherwise. The optimal soft information used by the decoder is the a posteriori log likelihood ratio

$$L(b_i|\boldsymbol{y}) \triangleq \log\left(\frac{p(b_i=1|\boldsymbol{y})}{p(b_i=0|\boldsymbol{y})}\right), \tag{2}$$

where b_i is the i:th bit of the transmitted symbol vector $\boldsymbol{s}$. The ratio (2) is known as soft-output and it tells us to which extent b_i is likely to be 0 and 1, given the received data $\boldsymbol{y}$.

1) Soft Maximum Likelihood: By using Bayes' rule and assuming uniform a priori probabilities in (2), we get the log likelihood ratio (LLR)

$$L(b_i|\boldsymbol{y}) = \log\left(\frac{\sum_{\boldsymbol{s}:b_i(\boldsymbol{s})=1}\exp\left(-\frac{1}{N_0}\left\|\boldsymbol{y}-\boldsymbol{H}\boldsymbol{s}\right\|^2\right)}{\sum_{\boldsymbol{s}:b_i(\boldsymbol{s})=0}\exp\left(-\frac{1}{N_0}\left\|\boldsymbol{y}-\boldsymbol{H}\boldsymbol{s}\right\|^2\right)}\right) \tag{3}$$

where the notation $\sum_{\boldsymbol{s}:b_i(\boldsymbol{s})=b}$ means the sum over all symbol vectors $\boldsymbol{s}$ of which the i:th bit is equal to b.

The major problem with evaluating (3) is that the sums contain 2^{N_T} terms, which gives rise to high computational complexity. Therefore, methods that do not have exponential complexity in N_T, such as the PM method, must be employed.

2) Soft Zero Forcing: The ZF detector is only applicable when $\boldsymbol{H}$ has full column-rank, which for randomly generated $\boldsymbol{H}$ is true with probability one. The ZF detector can be expressed as N_T parallel additive white Gaussian noise (AWGN) channels. More precisely,

$$\hat{\boldsymbol{s}}_{\mathrm{ZF}} \triangleq (\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{y} = \boldsymbol{s} + \underbrace{(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{e}}_{\tilde{\boldsymbol{e}}},$$

where each symbol s_i (i:th element of $\boldsymbol{s}$) is assumed to be transmitted through the AWGN channel,

$$\hat{s}_{\mathrm{ZF},i} = s_i + \tilde{e}_i, \quad \tilde{e}_i \sim \mathcal{N}\left(0, \frac{N_0}{2}(\boldsymbol{H}^T\boldsymbol{H})^{-1}_{i,i}\right), \tag{4}$$

where the noise terms $\tilde{e}_i$ are assumed to be uncorrelated over the index i. These channels can then be used to derive the soft ZF decisions $L(b_i|\hat{s}_{\mathrm{ZF},i})$ from (3),

$$L(b_i|\hat{s}_{\mathrm{ZF},i}) = \log\left(\frac{\exp\left(\frac{-(\hat{s}_{\mathrm{ZF},i}-1)^2}{N_0(\boldsymbol{H}^T\boldsymbol{H})^{-1}_{i,i}}\right)}{\exp\left(\frac{-(\hat{s}_{\mathrm{ZF},i}+1)^2}{N_0(\boldsymbol{H}^T\boldsymbol{H})^{-1}_{i,i}}\right)}\right) = \frac{4\hat{s}_{\mathrm{ZF},i}}{N_0(\boldsymbol{H}^T\boldsymbol{H})^{-1}_{i,i}}. \tag{5}$$

The ZF complexity is cubic with respect to N_T. The soft ZF method is computationally more efficient than soft ML, but it performs poorly for ill-conditioned matrices.

3) The Soft-Output via Partial Marginalization Method: The PM method in [6] and [7] offers a trade-off, by adjusting a parameter $r \in \{0,\ldots,N_\mathrm{T}-1\}$, between exact and approximate computation of (3). We present the slightly modified version in [7] of the method in [6]. The modified version in [7] has a somewhat simpler structure than the one in [6] without compromising any advantages of the original method in [6]. In order to explain PM, we use the following partitioning

$$\boldsymbol{H}\boldsymbol{s} = \underbrace{\begin{bmatrix}\bar{\boldsymbol{H}} & \tilde{\boldsymbol{H}}\end{bmatrix}}_{\text{col. permut. of } \boldsymbol{H}} \underbrace{\begin{bmatrix}\bar{\boldsymbol{s}}^T & \tilde{\boldsymbol{s}}^T\end{bmatrix}^T}_{\text{permut. of } \boldsymbol{s}} = \bar{\boldsymbol{H}}\bar{\boldsymbol{s}} + \tilde{\boldsymbol{H}}\tilde{\boldsymbol{s}},$$

where $\bar{\boldsymbol{H}} \in \mathbb{R}^{N_\mathrm{R}\times(r+1)}$, $\tilde{\boldsymbol{H}} \in \mathbb{R}^{N_\mathrm{R}\times(N_\mathrm{T}-r-1)}$, $\bar{\boldsymbol{s}} \in \mathcal{S}^{r+1}$ contains the bit of interest (the i:th bit in the original vector $\boldsymbol{s}$), and $\tilde{\boldsymbol{s}} \in \mathcal{S}^{N_\mathrm{T}-r-1}$. The permutation ordering is chosen with the aim to minimize the condition number of $\tilde{\boldsymbol{H}}$, see [6]. The PM method is obtained via a two-step approximation of (3): the sums in (3) associated with $\tilde{\boldsymbol{s}}$ are approximated with a maximization

$$L(b_i|\boldsymbol{y}) \approx \log\left(\frac{\sum_{\bar{\boldsymbol{s}}:b_i(\boldsymbol{s})=1}\max_{\tilde{\boldsymbol{s}}\in\mathcal{S}^{N_\mathrm{T}-r-1}}\exp\left(-\frac{1}{N_0}\left\|\boldsymbol{y}-\bar{\boldsymbol{H}}\bar{\boldsymbol{s}}-\tilde{\boldsymbol{H}}\tilde{\boldsymbol{s}}\right\|^2\right)}{\sum_{\bar{\boldsymbol{s}}:b_i(\boldsymbol{s})=0}\max_{\tilde{\boldsymbol{s}}\in\mathcal{S}^{N_\mathrm{T}-r-1}}\exp\left(-\frac{1}{N_0}\left\|\boldsymbol{y}-\bar{\boldsymbol{H}}\bar{\boldsymbol{s}}-\tilde{\boldsymbol{H}}\tilde{\boldsymbol{s}}\right\|^2\right)}\right), \tag{6}$$

and then this maximization is approximated with a sub-optimal hard linear detector, see [6]. We denote the PM detector outputs with $l(b_i|\boldsymbol{y})$. Hence $l(b_i|\boldsymbol{y})$ refers to the right-hand side of (6) where the maximizations are approximated via a linear detector. In the simplest form, the sub-optimal linear detector can be the hard ZF detector, which is well-known and well-studied. The hard ZF method is computationally more efficient than exact maximization, but it performs poorly for ill-conditioned matrices. However, the maximization problems in (6) are generally well-conditioned since the matrices $\tilde{\boldsymbol{H}}$ are tall. Notably, PM performs soft ZF for $r = 0$ and soft ML (exact LLR) for $r = N_\mathrm{T} - 1$. The complexity of PM with respect to the r-parameter is roughly $\mathcal{O}(\gamma 2^r)$ where $\gamma > 0$ is a constant, see App. A.

III. Proposed Allocation Framework

We are interested in allocating computational resources over multiple independent channel realizations that a codeword has propagated through, under a total per-codeword complexity constraint. A baseline approach would be to set a predetermined fixed distribution of the computational resources

independently of these channel realizations. Our goal is to find an on-line detection (demodulation) rule as a function of these matrices. We want to do this dynamically by adjusting the trade-off parameter of each detector in Fig. 1.

A. General Formulation

We formulate the optimization problem of interest in a general form as

$$\begin{aligned}\{r_1^*,\ldots,r_K^*\} = &\underset{\{r_1,\ldots,r_K\}}{\operatorname{argmax}} \sum_{k=1}^{K} D(r_k,\boldsymbol{H}_k)\\ &\text{subject to } \sum_{k=1}^{K} C(r_k,\boldsymbol{H}_k) \le C_{\text{tot}},\\ &\qquad r_k \in \{0,1,\ldots,\alpha\},\end{aligned} \tag{7}$$

where we have reintroduced the block index k in order to differentiate between the different channel matrices. The parameter $r_k \in \{0,1,\ldots,\alpha\}$ denotes for some integer α the complexity level used for a particular realization $\boldsymbol{H}_k$, the function $C(r_k,\boldsymbol{H}_k)$ is the computational complexity associated with r_k and $\boldsymbol{H}_k$, the performance measure $D(r_k,\boldsymbol{H}_k)$ quantifies the corresponding expected detection performance, and C_{tot} is the total computational complexity permitted per codeword. It is natural and necessary that the functions $D(r_k,\boldsymbol{H}_k)$ and $C(r_k,\boldsymbol{H}_k)$ are monotonically increasing with respect to the complexity level parameter r_k. Otherwise, the optimization problem would be impractical since it would mean that either the complexity (cost) or the detection performance is decreasing with increasing r_k.

In general, it is difficult to solve (7) in closed form since the expressions of the functions $D(r_k,\boldsymbol{H}_k)$ and $C(r_k,\boldsymbol{H}_k)$ can be very complicated. Furthermore, the optimization problem is combinatorial with $(\alpha+1)^K$ possible solution candidates and an optimal brute-force solution is typically not computationally feasible. However, we present several different approaches for solving this depending on what assumptions are made.

B. Prediction Of Detector Performance

The described problem of interest requires that we can, for a given noise level, channel realization $\boldsymbol{H}_k$, and complexity level r_k, quantify how well the detector is expected to perform in terms of frame error-rate (FER) and amount of involved computations. Further, it is also required that these quantities are separable for different $\boldsymbol{H}_k$. For some detectors, identifying such quantities is difficult.

The FER would be the ultimate performance measure since it is used to determine the actual performance. However, FER cannot be measured per channel $\boldsymbol{H}_k$. Additionally, the FER cannot be measured without explicitly running the whole communication chain for all channels of a frame, which is not feasible. Therefore, it would be more appropriately to quantify the performance by predicting the information loss per channel $\boldsymbol{H}_k$ from the encoder outputs to the corresponding decoder inputs. Theoretically, for infinitely long codes, the mutual information (MI) for a specific $\boldsymbol{H}_k$ between the transmitted signal vector (encoder outputs) $\boldsymbol{s}$ and the detected soft vector (decoder inputs) $\boldsymbol{l}(\boldsymbol{y}) = [l(b_1|\boldsymbol{y})\ l(b_2|\boldsymbol{y})\ \ldots\ l(b_{N_\text{T}}|\boldsymbol{y})]^T$, accumulated over all detected vectors, predicts whether the decoder will receive enough information or not in order to succeed. This MI per channel $\boldsymbol{H}_k$ is defined as

$$\begin{aligned}I(\boldsymbol{l}(\boldsymbol{y});\boldsymbol{s}) &\triangleq \sum_{\boldsymbol{s}} \int_{\mathbb{R}^{N_\text{T}}} p(\boldsymbol{l}(\boldsymbol{y}),\boldsymbol{s}) \log_2\left(\frac{p(\boldsymbol{l}(\boldsymbol{y}),\boldsymbol{s})}{p(\boldsymbol{l}(\boldsymbol{y}))p(\boldsymbol{s})}\right) d\boldsymbol{l}(\boldsymbol{y})\\ &= \sum_{\boldsymbol{s}} \int_{\mathbb{R}^{N_\text{T}}} \frac{p(\boldsymbol{l}(\boldsymbol{y})|\boldsymbol{s})}{|\mathcal{S}|^{N_\text{T}}} \log_2\left(\frac{|\mathcal{S}|^{N_\text{T}} p(\boldsymbol{l}(\boldsymbol{y})|\boldsymbol{s})}{\sum_{\boldsymbol{s}'} p(\boldsymbol{l}(\boldsymbol{y})|\boldsymbol{s}')}\right) d\boldsymbol{l}(\boldsymbol{y}).\end{aligned} \tag{8}$$

In BICM systems, the detected bits are assumed to be mutually independent given the received vector $\boldsymbol{y}$, see [15]. Therefore, in such a setting with a BPSK constellation, we want to measure the MI for a specific $\boldsymbol{H}_k$ between encoder outputs and decoder inputs defined as

$$I_{\text{BICM}}(\boldsymbol{l}(\boldsymbol{y});\boldsymbol{s}) \triangleq \sum_{i=1}^{N_\text{T}} I(l(b_i|\boldsymbol{y});b_i). \tag{9}$$

In the simplistic case with binary signaling ($b \in \{0,1\}$) when we can find an AWGN relation $L(b|\boldsymbol{y}) = a(b) + n$ where $n \sim \mathcal{N}(0,\sigma_n^2)$, the MI $I(L(b|\boldsymbol{y});b)$ becomes

$$\begin{aligned}I_{\text{AWGN}}(L(b|\boldsymbol{y});b) \triangleq 1 - &\frac{1}{2\sqrt{2\pi\sigma_n^2}} \int_{\mathbb{R}} \exp\left(-\frac{n^2}{2\sigma_n^2}\right)\\ &\times \sum_{b\in\{0,1\}} \log_2\left(1+\exp\left(\frac{n^2-(n-2a(b))^2}{2\sigma_n^2}\right)\right) dn.\end{aligned} \tag{10}$$

For the BICM MI of ZF for a specific $\boldsymbol{H}_k$, the MI values $I(L(b_i|\hat{s}_{\text{ZF},i});b_i)$ are computed first; this is done by using the expression in (10) applied to the symbol-wise equivalent AWGN channel given by (5). Then, the so-obtained MI values $I(L(b_i|\hat{s}_{\text{ZF},i});b_i)$ are summed up in (9).

To find the analytical solution of the integrals in (8), (9), and (10) is non-trivial. In addition, the conditional probability densities $p(\boldsymbol{l}(\boldsymbol{y})|\boldsymbol{s})$ in (8) and $p(l(b_i|\boldsymbol{y})|b_i)$ in (9) are generally not known and very difficult to acquire analytically. Empirical evaluation of the integral in (10) is straightforward since the integrand is known and one-dimensional. This is however not the case for the integrals in (8) and (9). Moreover, empirical evaluations of the integral and the distribution $p(\boldsymbol{l}(\boldsymbol{y})|\boldsymbol{s})$ in (8) are substantially more demanding computationally than the corresponding evaluation in (9), as (8) involves multidimensional integration whereas (9) consists of one-dimensional integrals and distributions. Recall that this evaluation must be done per channel matrix $\boldsymbol{H}_k$, which makes matters even more difficult. In what follows, we investigate different techniques to estimate these MI values.

1) Empirical MI Evaluation: One way to acquire MI values is through off-line empirical evaluations. Primarily, these values are suitable to benchmark the proposed approximations of the MI and to investigate what the best achievable FER is when accurate MI values are used. We have tabulated a large number of empirical MI values between encoder outputs and decoder inputs (PM detector outputs $\boldsymbol{l}(\boldsymbol{y}) = [l(b_1|\boldsymbol{y})\ l(b_2|\boldsymbol{y})\ \ldots\ l(b_{N_\text{T}}|\boldsymbol{y})]^T$), both for $I(\boldsymbol{l}(\boldsymbol{y})|\boldsymbol{s})$ and $I_{\text{BICM}}(\boldsymbol{l}(\boldsymbol{y})|\boldsymbol{s})$, each evaluated for a randomly generated MIMO matrix. This has been done by first finding empirically

the conditional distributions $p(\boldsymbol{l}(\boldsymbol{y})|\boldsymbol{s})$ in (8) and $p(l(b_i|\boldsymbol{y})|b_i)$ in (9), and then evaluating empirically the integrals in (8) and (9) for each such channel realization.

These tables could be utilized in practice for applications in need of MI values on-line. The difficulty is that the channels generated on-line will not be exactly the same as the channels stored off-line in the tables. However, these tables contain MI values that represent the whole MI span, i.e., in our setting from 0 to N_{T} bits. Additionally, with a proper indexing function, the storage of the table can be kept reasonable. Therefore, making such tables practical is a matter of finding appropriate indexing functions, which in addition need to be evaluated efficiently. The main issue is to identify the key properties of a channel matrix that influence the information flow. This is a very difficult task since the matrices are multidimensional elements that can have completely different values and still yield the same MI.

In the simple case with an AWGN channel as in (10), the noise variance σ_n^2 in (10) defines a one-to-one indexing function that completely describes the MI. Thus, for the AWGN case, a table with arbitrarily good resolution can be generated and the MI values can be accessed exactly, up to the accuracy of the particular table resolution. For the BICM MI of ZF, the ZF error variance in (4) can be used as the one-to-one indexing function.

In Fig. 2, we illustrate and motivate that the accumulated MI is an appropriate performance measure. Specifically, we show how well ZF BICM MI predicts whether the decoder will succeed or not in a simple BICM MIMO receiver with a soft ZF detector. This figure illustrates two distinct histograms of the MI of a whole frame: one when the decoder succeeds and one when it fails. The value where the histograms separate in Fig. 2 is approximately 5000 since the codewords in Fig. 2 contain approximately 5000 information bits. Note that the center of the region where the histograms overlap is not as important as the size of the region. The histograms in Fig. 2 intersect within a small region, which implies that the accumulated mutual information predicts the outcome (succeed or fail) of the decoder well.

2) Interpolation of MI for PM: For Gaussian signaling, we know the mutual information $I(\boldsymbol{l}(\boldsymbol{y});\boldsymbol{s})$ for the lowest complexity level $r_k = 0$ (ZF method) and the highest complexity level $r_k = N_{\mathrm{T}} - 1$ (ML method), respectively. In [16], it has been proven that the exact LLR is a sufficient statistic of $\boldsymbol{y}$ about $\boldsymbol{s}$, i.e., the ML method is information loss-less. Hence, with the channel model in (1) and a Gaussian constellation with $E[\boldsymbol{s}\boldsymbol{s}^T] = \frac{1}{2}\boldsymbol{I}$, the MI for the ML method is

$$\bar{I}(\boldsymbol{H}_k) \triangleq \log_2 \det\left(\boldsymbol{I} + \frac{\boldsymbol{H}_k \boldsymbol{H}_k^T}{N_0}\right). \tag{11}$$

For the ZF method, the corresponding quantity is

$$\tilde{I}(\boldsymbol{H}_k) \triangleq \sum_{i=1}^{N_{\mathrm{T}}} \bar{I}\left(\frac{1}{\sqrt{(\boldsymbol{H}_k^T \boldsymbol{H}_k)^{-1}_{(i,i)}}}\right). \tag{12}$$

These two expressions can be interpolated to approximate the objective function $D(r_k, \boldsymbol{H}_k)$ for any $r_k \in \{0, \ldots, N_{\mathrm{T}} - 1\}$.

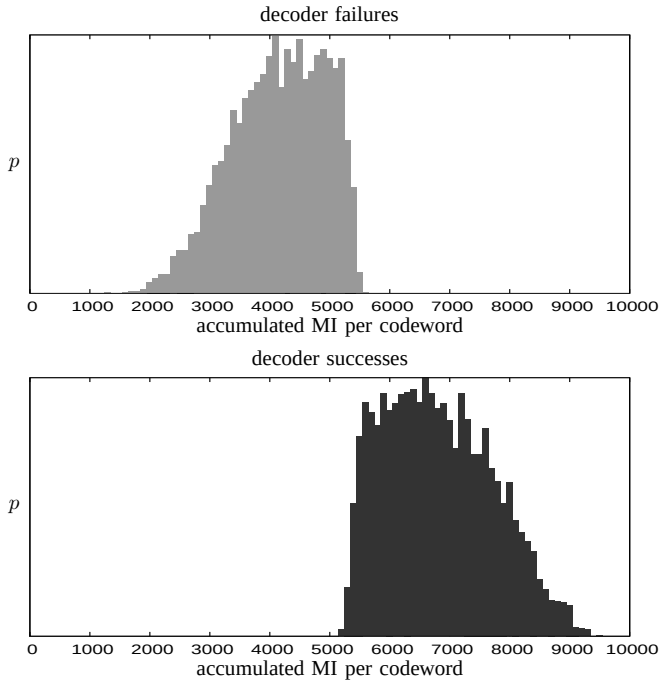


Fig. 2: MI histograms. We used a complex-valued 3×3-MIMO with QPSK and the soft ZF detector with a 10000 bits long LDPC rate $1/2$ code. The accumulated MI is a sum of MI values over a whole codeword. The upper histogram shows the distribution of MI when the decoder fails and the lower shows the opposite. Since the intersection of these histograms is small, the MI predicts whether the decoder will succeed or not reliably. The probability of this intersection is approximately 0.07.

In the simplest form a linear interpolation can be used

$$D(r_k, \boldsymbol{H}_k) \triangleq \frac{1}{N_{\mathrm{T}} - 1}((N_{\mathrm{T}} - 1 - r_k)\tilde{I}(\boldsymbol{H}_k) + r_k \bar{I}(\boldsymbol{H}_k)). \tag{13}$$

Note that this is a crude approximation. First, the actual signal constellation we use is BPSK modulated and therefore far from Gaussian as assumed by (11) and (12). It is possible that, in an application with adaptive modulation and coding, the performance measure in (13) could be extended to take into account different modulation schemes chosen by the transmitter. Second, the MI is not necessarily linear with respect to r_k. Nevertheless, this approximation yields good results, as we will see later on, and is very practical since it does not require much storage nor heavy computations.

C. Allocation Approaches

1) Optimal Solution with Dynamic Programming: The problem in (7) can be formulated as a dynamic programming problem and a trellis structure can be acquired, which we can solve optimally, see [17]. A very important advantage of this method is that optimality holds for any well-defined objective function $D(r_k, \boldsymbol{H}_k)$ and cost function $C(r_k, \boldsymbol{H}_k)$. Note that the optimality claim holds for the particular objective and cost function that is used, which does not necessarily yield optimality in the FER sense.

In dynamic programming, there are three important definitions: the steps, the states, and the decision rules. For our problem in (7), the number of steps is equal to the number

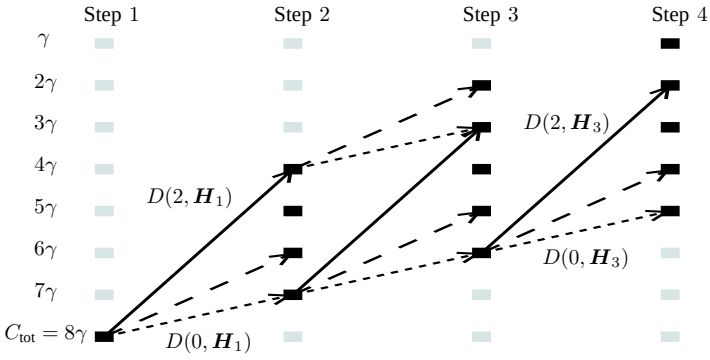


Fig. 3: Illustration of the trellis structure. In this example, we let the number of channel matrices $K = 4$. The PM complexity function $C(r, \boldsymbol{H}) = \gamma 2^r$ with $C_{\text{tot}} = 4\gamma 2^1 = 8\gamma$ is used, which yields that each state $c_1, \ldots, c_4 \in \{\gamma, \ldots, 8\gamma\}$. The decisions (steps) $r_1, \ldots, r_4$ are each restricted to be within $\{0, 1, 2\}$. The decision candidates in step k are represented by different lines: solid for $r_k = 2$, long-dashed for $r_k = 1$, and short-dashed for $r_k = 0$. The branch metric is $D(r_k, \boldsymbol{H}_k)$ and the branches in step 4 are not visible.

of channel realizations per codeword, the state variable c_k is the complexity budget left from step k to K, and the decision variable r_k is the complexity level to be determined in step k. Using these definitions and backward recursion, we can define the state transition from c_k to $c_{k+1} = c_k - C(r_k, H_k)$. The accumulated metric that we want to maximize is thus defined as

$$\begin{aligned} f_k(c_k) &\triangleq \max_{r_k}\{D(r_k, \boldsymbol{H}_k) + f_{k+1}(c_{k+1})\} \\ &= \max_{r_k}\{D(r_k, \boldsymbol{H}_k) + f_{k+1}(c_k - C(r_k, \boldsymbol{H}_k))\}, \end{aligned}$$

where with backward recursion, the initial values are $f_{K+1} = 0$ and $c_1 = C_{\text{tot}}$. Making sure in each step that the constraints in (7) are fulfilled will yield the resulting optimal solution $f_1(c_1)$. Pictorially, this algorithm is illustrated in Fig. 3. The complexity of the algorithm itself is decided by the number of possible trellis points that can be visited. The number of trellis points is given by the number of decisions that need to be made and the number of different values that the states can take on. Since we exemplify our ideas with the PM method, which has the complexity function $C(r_k, \boldsymbol{H}_k) = \gamma 2^{r_k}$, the number of possible state values is $\left\lfloor \frac{C_{\text{tot}}}{\gamma} \right\rfloor$. This number is usually a multiple of K since C_{tot} is generally decided a priori based on the number of channel matrices per codeword. For instance, in our setting, $C_{\text{tot}} = K\gamma 2^{r_{\text{fix}}}$ for some fix r_{fix} value. Another practicality that can be taken into account is that detection with the complexity level r_k above a certain maximum value is computationally unfeasible. For instance, we can restrict $r_k \in \{0, 1, 2\}$, which would yield the total worst-case complexity of this algorithm as $\mathcal{O}(K^2)$.

2) Sub-Optimal Greedy Solution: We propose a simple greedy heuristic in Alg. 1 that adaptively allocates the limited computational resources for different channel realizations. It is greedy since it deals out more complexity to the detection problems that give the largest performance gain. The idea is to start with a resource budget, which in our case is the complexity at hand C_{tot}. Initially, the corresponding parameters r_k are set to zero, which is equivalent to performing detection with the lowest complexity level on all matrices. In each iteration step of the algorithm, we identify the matrix $\boldsymbol{H}_{\tilde{k}}$ that yields the largest performance gain $\Delta D(r_{\tilde{k}}, \boldsymbol{H}_{\tilde{k}})$ defined as

$$\Delta D(r, \boldsymbol{H}) \triangleq D(r+1, \boldsymbol{H}) - D(r, \boldsymbol{H}).$$

Subsequently, the corresponding $r_{\tilde{k}}$ is incremented whereas the resource budget is reduced by $\Delta C(r_{\tilde{k}}, \boldsymbol{H}_{\tilde{k}})$, where

$$\Delta C(r, \boldsymbol{H}) \triangleq C(r, \boldsymbol{H}) - C(r-1, \boldsymbol{H}).$$

Now, the errors of the detection problems related to matrix $\boldsymbol{H}_{\tilde{k}}$ are smaller than previously. The procedure is repeated until we have emptied our resource budget. The steps of this heuristic method are summarized in Alg. 1.

Algorithm 1 Greedy heuristic algorithm for allocation

Start with $\{\boldsymbol{H}_1, \ldots, \boldsymbol{H}_K\}$, $r_k := 0$ and
$\Delta D_k := \Delta D(0, \boldsymbol{H}_k)$ for all $k \in \{1, \ldots, K\}$,
and $C_{\mathsf{bag}} := C_{\text{tot}}$.
while $C_{\mathsf{bag}} > 0$ **do**
  $\tilde{k} := \arg\max_k \{\Delta D_k\}_{k=1}^K$
  if $r_{\tilde{k}} < \alpha$ and $\Delta C(r_{\tilde{k}} + 1, \boldsymbol{H}_{\tilde{k}}) \leq C_{\mathsf{bag}}$ **then**
    $r_{\tilde{k}} := r_{\tilde{k}} + 1$
    $\Delta D_{\tilde{k}} := \Delta D(r_{\tilde{k}}, \boldsymbol{H}_{\tilde{k}})$
    $C_{\mathsf{bag}} := C_{\mathsf{bag}} - \Delta C(r_{\tilde{k}}, \boldsymbol{H}_{\tilde{k}})$
  else
    Keep $r_{\tilde{k}}$ as is and ignore the $\tilde{k}$:th
    allocation problem from now on
    by setting $\Delta D_{\tilde{k}} = -\infty$.
  end if
end while

To make the maximization of each iteration in Alg. 1 easy, we initiate by sorting the ΔD_k values in Alg. 1, requiring $K \log(K)$ operations. Then, for each iteration, the updated $\Delta D_{\tilde{k}}$ value is sorted, which requires $\log(K)$ operations. The number of iterations depends on the total available per-codeword complexity C_{tot}, which as mentioned in Sec. III-C1 is generally a multiple of K. In our setting for instance, $C_{\text{tot}} = K\gamma 2^{r_{\text{fix}}}$ for some fix r_{fix} value. Hence, the worst-case complexity of this greedy heuristic is $\mathcal{O}(K \log(K))$.

3) Sub-Optimal Solution with Waterfilling: To solve the optimization problem in (7) analytically, we relax $r_k \in \{0, \ldots, N_{\text{T}} - 1\}$ to a continuous variable $\tilde{r}_k \geq 0$ and then approximate $D(\tilde{r}_k, \boldsymbol{H}_k)$ with the linear interpolation of $\tilde{I}(\boldsymbol{H}_k)$ and $\bar{I}(\boldsymbol{H}_k)$ in (13). We use this to formulate a simple convex optimization problem, which can be solved optimally using the Lagrangian multiplier $\lambda \geq 0$ as follows,

$$\begin{aligned} &\mathcal{L}(\tilde{r}_1, \ldots, \tilde{r}_K, \lambda) \triangleq \sum_{k=1}^{K} D(\tilde{r}_k, \boldsymbol{H}_k) + \lambda(C_{\text{tot}} - \gamma 2^{\tilde{r}_k}), \\ &\frac{\partial \mathcal{L}(\tilde{r}_1, \ldots, \tilde{r}_K, \lambda)}{\partial \tilde{r}_k} = \frac{\bar{I}(\boldsymbol{H}_k) - \tilde{I}(\boldsymbol{H}_k)}{N_{\text{T}} - 1} - \lambda \log(2) \gamma 2^{\tilde{r}_k} = 0 \\ &\implies \tilde{r}_k = \underbrace{\log_2\left(\frac{1}{\lambda}\right)}_{\triangleq \nu} - \underbrace{\log_2\left(\frac{(N_{\text{T}} - 1)\log(2)\gamma}{\bar{I}(\boldsymbol{H}_k) - \tilde{I}(\boldsymbol{H}_k)}\right)}_{\triangleq \beta_k}. \end{aligned}$$

The optimal non-integer solution is given by

$$\tilde{r}_k^* = \begin{cases} \nu^* - \beta_k, & \nu^* > \beta_k \\ 0, & \nu^* \leq \beta_k \end{cases},$$

where ν^* is chosen such that $\sum_{k=1}^{K} \gamma 2^{\tilde{r}_k^*} = C_{\text{tot}}$. This calculation is reminiscent to the waterfilling technique known in information theory, see [1, p. 183]. The resulting integer solution r_k^* is acquired by flooring the respective $\tilde{r}_k^*$ and then increasing r_k^* step-wise for which the difference $\tilde{r}_k^* - r_k^*$ is largest, until we can not increase any r_k^* without violating the complexity constraint. With this technique, the resulting integer values r_k^* should most likely be close to the optimal integer solution. The worst-case complexity of this algorithm is well-known to be $\mathcal{O}(K^2)$.

IV. Numerical Experiments

To illustrate the results, we have used Monte Carlo simulation to obtain the FER as a function of E_b/N_0, where E_b is the energy per information bit. We have used QPSK modulation with a 3×3 and a 6×6 complex MIMO system with i.i.d. $\mathcal{CN}(0, 1)$ elements. The detection is however performed on a real-valued 6×6 and 12×12 MIMO system with a BPSK constellation, which is done by decomposing the channel matrix as mentioned in Sec. II-A. We have used data packets consisting of roughly 10000 coded bits encoded with LDPC and bit-interleaved convolutional codes.

A. MI Table Generation

1) AWGN MI: It was straightforward to evaluate the integral in (10) empirically using Monte Carlo integration since the distribution from which to draw samples is known (Gaussian), the integrand is known, and the integral is one-dimensional. These MI values were generated and stored for 128000 different values of σ_n in (10) that cover the whole MI span between 0 and 1 bits. Each of the stored MI values were computed with a standard deviation below 0.01 bits per channel use.

2) PM MI & PM BICM MI: To generate the tables for the MI of PM and the BICM MI of PM, we used Gaussian mixture models (GMM) to estimate the distributions $p(\boldsymbol{l}(\boldsymbol{y})|\boldsymbol{s})$ in (8) and $p(l(b_i|\boldsymbol{y})|b_i)$ in (9) with 7 and 3 Gaussian modes, respectively. Then, we used Monte Carlo integration to empirically compute the MI integrals in (8) and (9) with standard deviations below 0.025 bits. We constructed these tables with the $1/N_0$ values spanning from -4 dB to 10 dB for the 3×3 complex-valued MIMO channel with QPSK specified in Sec. IV. For each such channel realization, the MI values were evaluated for three different complexity levels $r \in \{0, 1, 2\}$, which we know a priori from initial experiments are the values that are of interest in our setting. The tables were built from 10000 channel matrices per $1/N_0$ value, which is in total 150000 channel matrices and required approximately 90000 core-hours altogether to establish. The tables are available from the authors.

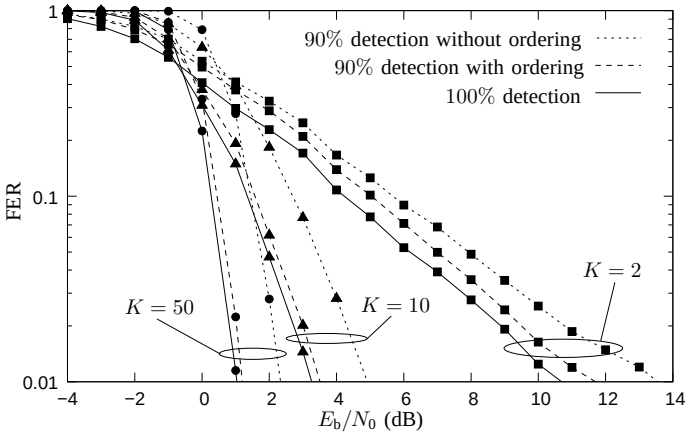


Fig. 4: Performance comparison. We have used the soft ZF detector. The plots are divided into three groups, each with a fixed coherence time that corresponds to $K \in \{50, 10, 2\}$ independent channels per codeword. Each such group contains plots of three different detection strategies: detecting 90% of the received vectors on a first-in first-detected basis, detecting 90% of the received vectors by ordering them properly, and detecting 100% of the received vectors. These plots show that proper ordering yields significant performance gains, especially for small coherence times (large K) where the performance is close to that of detecting 100% of the frame.

B. Results

1) The Special Case of Binary Decisions: As mentioned in Sec. I, we know that it is not necessary in a coded system to correctly detect all the bits in a codeword. If the error correcting code is strong enough, then it can handle some erased bits. The LLRs for the erased bits would then be set to zero. If only a portion of the bits in a codeword are detected, which portion should we detect and which should we skip in order to minimize the FER, given a predetermined maximal allowed per-codeword complexity?

This binary example (detect or skip) is a special case of the general problem in Sec. III-A since it can be deduced from (7) by letting $r_k \in \{0, 1\}$ where 0 means skip and 1 means detect. The optimization in (7) is simply solved by ordering the symbols/matrices according to the corresponding $D(1, \boldsymbol{H}_k)$ values. In this section we only consider the ZF detector. The reason is that the output of this (linear) detector can be, in BICM settings, expressed as the output of N_T parallel AWGN channels, for which we can accurately and easily compute the MI values. We take $D(r_k, \boldsymbol{H}_k)$ in (7) to be the ZF BICM MI, which is efficiently evaluated through a table lookup for any $\boldsymbol{H}_k$ using the AWGN channels in (4) and the AWGN MI table in Sec. IV-A1. In this example we assume that 90% of the codeword is detected, whereas the rest is disregarded and the corresponding bit erasures are handled by the decoder.

The performance results are shown in Fig. 4 for randomly generated 3×3 complex-valued MIMO matrices. These results show how, by simple means, substantial improvements can be achieved in a practical system. In particular, this is the case when the channel varies often so that K is large. The reason is that very small computational efforts $\mathcal{O}(K \log(K))$ are required to determine the ordering and that linear methods work well in such (large K) scenarios, see [1, p. 362].

2) The General Multi-Decision Scenario with PM: Next we exemplify how the proposed allocation framework can be used

in a multi-decision scenario, using the PM detector. In order to include the methods that rely on accurate MI values in the comparison, all simulations with $N_{\mathrm{R}} = N_{\mathrm{T}} = 6$ were carried out using only the channels that are available in the tables described in Sec. IV-A2. We pick the channels at random for each simulated frame. The probability of getting the same combination of channels in two codewords is $1/\binom{10^4}{K}$ so the results should be statistically reliable.

The channels are assumed to be block fading with different numbers of symbol vectors per independent matrix realization (coherence times) $L \in \{85, 150, 260\}$ for different illustrations. The number of independent channel matrix realizations per data packet (frame) is $K = \left\lceil \frac{10000}{L \times N_{\mathrm{T}}} \right\rceil \in \{20, 7, 6\}$ in the following simulations. We compare the performance of our allocation schemes with that of the baseline approach with uniform allocation. The per-codeword complexity $C_{\text{bgt}} = K\gamma 2^{r_{\text{fix}}} = 2K\gamma$ assigned by the baseline approach for $r_{\text{fix}} = 1$ is used to limit the total per-codeword available complexity C_{tot}. We use four allocation schemes:

- **D-MMI**: Optimal solution using dynamic programming in Sec. III-C1 with the objective function $D(r_k, \boldsymbol{H}_k)$ as the MIMO MI in (8).
- **D-BMI**: Optimal solution using dynamic programming in Sec. III-C1 with the objective function $D(r_k, \boldsymbol{H}_k)$ as the BICM MI in (9).
- **G-BMI**: Sub-optimal solution using the greedy method in Sec. III-C2 with the objective function $D(r_k, \boldsymbol{H}_k)$ as the BICM MI in (9).
- **W-LMI**: Sub-optimal solution using the waterfilling technique in Sec. III-C3 with the objective function $D(r_k, \boldsymbol{H}_k)$ as the MI approximated by the linear interpolation in (13).

Note that the MI tables described in Sec. IV-A2 are only needed for the MI values of the listed first three schemes. The MI values of the fourth scheme (W-LMI) can be efficiently computed for any arbitrary channel matrix. The performance results in terms of FER are presented in Tab. 1, and in Fig. 5, 6, 7, and 8.

In Fig. 5 we plot the comparisons for $N_{\mathrm{R}} = N_{\mathrm{T}} = 6$ and $L = 260$ ($K = 7$) using different codes: LDPC code with rate $1/2$ and $2/3$ in Fig. 5(a) and Fig. 5(c) respectively, and bit-interleaved convolutional code with rate $1/3$ and $1/2$ in Fig. 5(b) and Fig. 5(d) respectively. In Fig. 5, we have also used a "Genie" allocator that minimizes the FER optimally by running the whole communication chain brute-force and choosing the resource distribution that yields the smallest FER.

Tab. 1 summarizes the main results of Fig. 5. The values in Tab. 1 were obtained by reading from Fig. 5. This table shows the gains in SNR of the W-LMI allocation scheme compared to uniform allocation for the codes used in Fig. 5 at 1% FER. We can observe in Tab. 1 that the choice of code affects the performance gain and that for strong codes (LDPC codes with low rates), the gain is less significant. This result follows due to the fact that stronger codes reduce the performance gap between poor and good detectors.

It is clear from Fig. 5 that the performance gain using any of the proposed multi-decision allocation schemes is substantial

	Conv. $1/2$	Conv. $1/3$	LDPC $2/3$	LDPC $1/2$
SNR Gain	3.9 dB	2.2 dB	0.55 dB	0.25 dB

Tab. 1: Performance comparison at fixed 1% FER. The table shows SNR gains of the practical scheme W-LMI with respect to uniform allocation with $r = 1$ for different codes using $N_{\mathrm{R}} = N_{\mathrm{T}} = 6$ and $L = 260$ ($K = 7$). This table summarizes the detailed curves in Fig. 5.

in comparison to the performance of the baseline approach. At low SNR, the performance gain is smaller than at higher SNR. This result follows from the fact that ML and ZF perform similarly at low SNR, whereas at high SNR, ML performs much better than ZF due to the large diversity gap between the methods. The trend observed in Tab. 1 regarding the choice of code is seen in Fig. 5 for all detection methods used in this paper. The curves in Fig. 5 show that the mutual information in (8) is not as good performance measure as the BICM MI expression in (9). This comes as no surprise since we are simulating a BICM setup. We can also see that the MI approximation using linear interpolation in (13) is a good approximation for the optimization problem of interest. This very simple approximation is efficiently evaluated and well-suited for practical implementations.

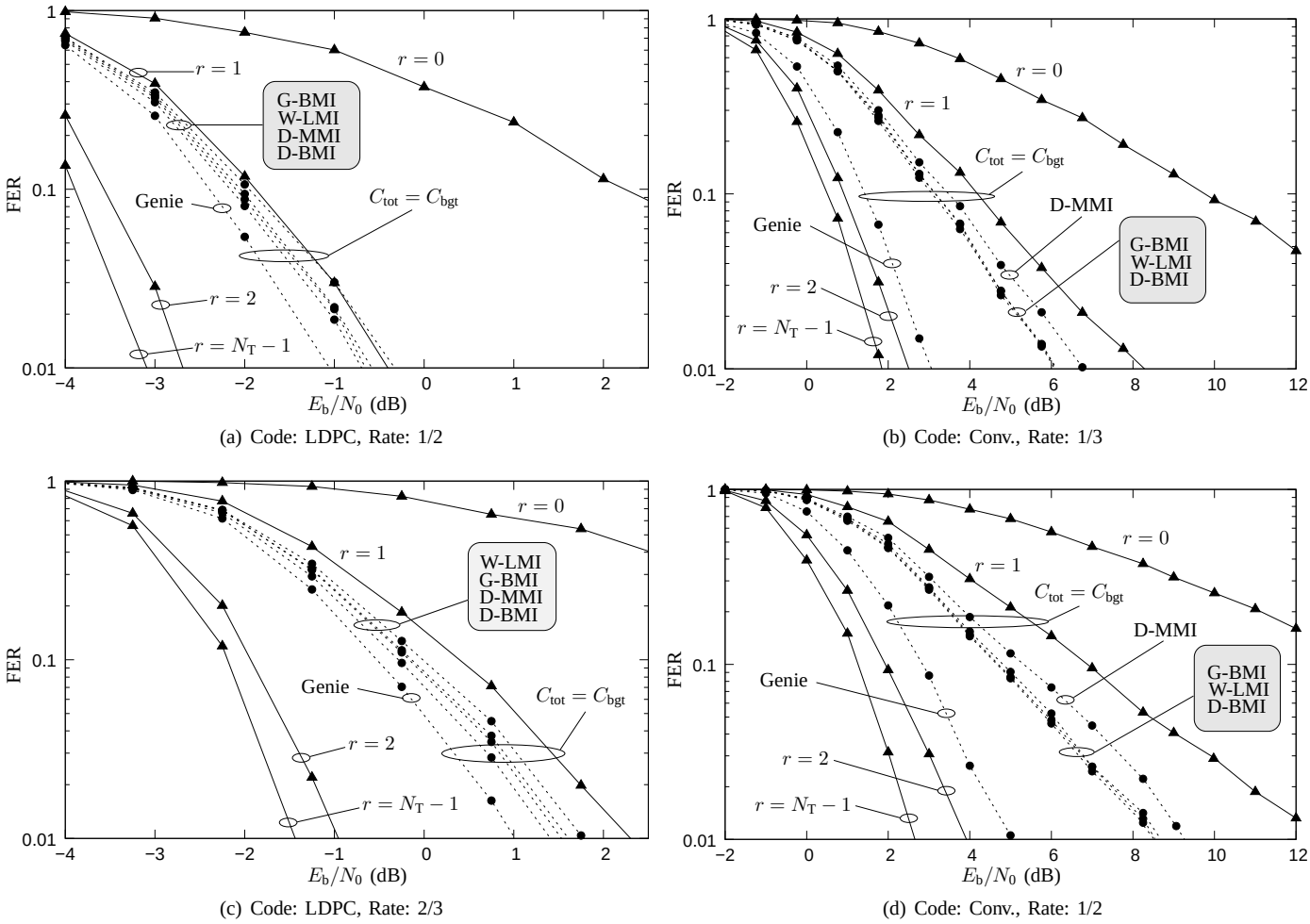
In Fig. 6, we illustrate how the performance of our practically implementable allocation scheme W-LMI varies for a wider range of C_{tot} using the same setting as in Fig. 5(d). In Fig. 6, by comparing the dashed lines ($0.75C_{\text{bgt}}$ and $1.5C_{\text{bgt}}$) with the solid lines (C_{bgt} and $2C_{\text{bgt}}$), we can see how 25% of the computations can be saved by using computational resource allocation.

In Fig. 7, we plot the performance for the same setting as in Fig. 5(d) except that the coherence time is decreased to $L = 85$ ($K = 20$). The same curves as in Fig. 5 are plotted except for the "Genie" allocator, which had to be excluded due to its massive complexity when $K = 20$. We can see that the performance gain is slightly larger in Fig. 7 compared to Fig. 5(d), which follows from the fact that the optimization in (7) has more degrees of freedom (larger K).

In Fig. 8, we plot the performance for the same setting as in Fig. 5(d) except that the number of antennas is increased to $N_{\mathrm{R}} = N_{\mathrm{T}} = 12$ and the coherence time is decreased to $L = 150$ ($K = 6$). We plot the W-LMI curve which does not require the MI tables in Sec. IV-A2, and the same solid-line curves as in Fig. 5, except for the PM with $r = N_{\mathrm{T}} - 1$ curve due to its complexity. We can see that the performance gain is similar to that in previous cases.

V. Discussion and Conclusions

We have presented a mathematical framework for optimized allocation of computational resources in soft MIMO detection, for the case where a codeword spans several channel realizations and a given per-codeword computational resource budget is available. In this framework, we have proposed several allocation algorithms that assign a complexity-tradeoff parameter to each individual detection problem to be solved. Our framework is particularly useful in scenarios when the channel matrices stay constant over several channel-uses but not over a whole codeword. Nevertheless, the extreme cases when the channel is either constant over a codeword, or



(a) Code: LDPC, Rate: 1/2

(b) Code: Conv., Rate: 1/3

(c) Code: LDPC, Rate: 2/3

(d) Code: Conv., Rate: 1/2

Fig. 5: Performance comparison. These plots were generated with $N_R = N_T = 6$ and $L = 260$, which corresponds to $K = 7$ independent channel realizations per codeword. Different lines represent different approaches of distributing the computational complexity budget. The solid lines with $r = x$ represent a uniform distribution (fixed r-parameter) over all matrices with the complexity levels set to x. The dashed lines represent proposed adaptive allocation schemes, specified in Sec. IV-B2, with the total complexity kept under C_{bgt}. The lines marked with a group of tags are denoted in such an order that tags from top to bottom correspond to lines from right to left.

varies from one received symbol vector to the next, can be handled by our framework as well. In open-loop settings, i.e., without any channel state information (CSI) at the transmitter, our techniques yield large performance gains as shown in Sec. IV. In closed-loop settings, we can expect our framework to be beneficial in cases with partial CSI at the transmitter. In cases where the transmitter has full CSI, it can perform beamforming along the right singular vectors of the channel matrix. MIMO detection would then become unnecessary and so would computational resource allocation consequently be. An interesting feature of our receiver architecture in closed-loop settings is that, since we are only treating channel bits, variable rates would not impose any difficulties when applying the proposed techniques.

For the simplistic, binary-decision scenario in Sec. IV-B1, we have shown that the performance can be drastically improved by simple means and with very small computational efforts. In the more general multi-decision scenario, our allocation schemes yielded performance gains up to 4 dB at 1% FER, see Tab. 1. Additionally, we have shown empirically that the W-LMI algorithm proposed in Sec. III-C3, which can be efficiently evaluated and is well-suited for practical implementations, performs close to the mutual-information-optimal D-BMI method proposed in Sec. III-C1. The worst-case complexity of these algorithms is $\mathcal{O}(K^2)$, which is nearly negligible in situations when $N_T \gtrsim K$ compared to the cost of performing the actual detection (even the simplest detectors such as ZF have a complexity $\mathcal{O}({N_T}^3)$). Additionally, in practical OFDM systems with coding across subcarriers, if the channel is frequency-selective and time-invariant so that the set of channel realizations seen by each codeword is the same for many consecutive codewords, the cost of performing the allocation may be amortized over several codewords.

Note also that in most scenarios, the value of K used may be smaller than the number of distinct channels supplied to the detectors. For example, if there is coding across subcarriers in OFDM, then the MIMO channels associated with neighboring subcarriers will be strongly correlated. For the purpose of computational resource allocation, subcarriers that have nearly the same channel matrix may then be grouped together and considered to have the same channel, whereas when performing the actual detection, the "true" channel

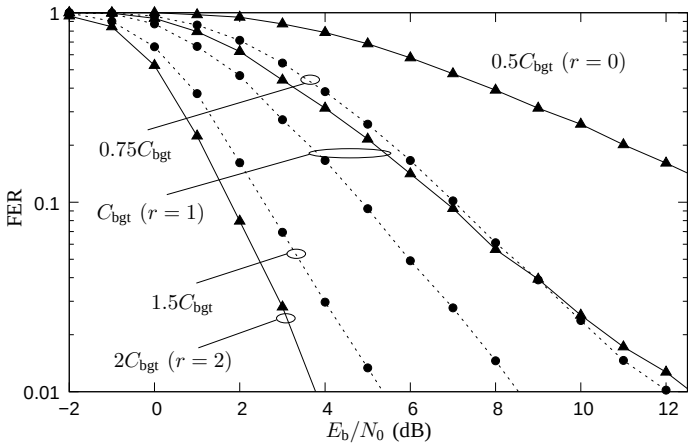


Fig. 6: Performance comparison. The same setting as in Fig. 5(d) is used but now with a wider complexity budget C_{tot} range. The dashed lines with circles show our practically implementable allocation scheme (W-LMI) using different $C_{\text{tot}} \in \{0.75C_{\text{bgt}}, C_{\text{bgt}}, 1.5C_{\text{bgt}}\}$. The solid lines with triangles show the baseline approach using uniform allocation for $C_{\text{tot}} \in \{0.5C_{\text{bgt}}, C_{\text{bgt}}, 2C_{\text{bgt}}\}$.

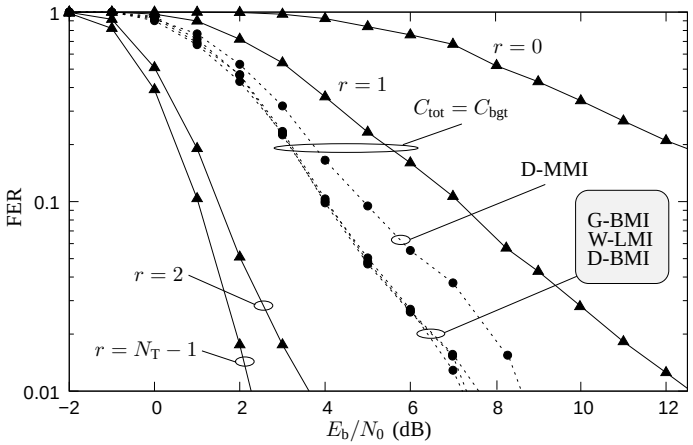


Fig. 7: Performance comparison. The same setting as in Fig. 5(d) is used except that the coherence time is $L = 85$ ($K = 20$). Further, the same curves are shown as in Fig. 5 except that the Genie allocator is excluded due to its massive complexity when $K = 20$. The description of the curves is the same as in Fig. 5.

matrix should be used for each subcarrier. This means that the parameter K in our proposed method, and in the complexity considerations, reflects the product of the channel coherence time and coherence bandwidth, rather than the number of different channel matrices supplied to the individual MIMO detectors.

Overall, the results indicate that it is indeed worth the effort to properly distribute the computational resources. The results of the Genie allocator in Fig. 5 show that there is much more left to gain if one could come up with a performance measure that predicts more accurately whether the decoder will succeed or not. This observation strongly motivates continued future investigations within the area of adaptive computational resource allocation for detection and decoding.

APPENDIX

A. PM Complexity

We consider a rough complexity count in terms of floating point operations (FLOPs) needed to evaluate the vectors

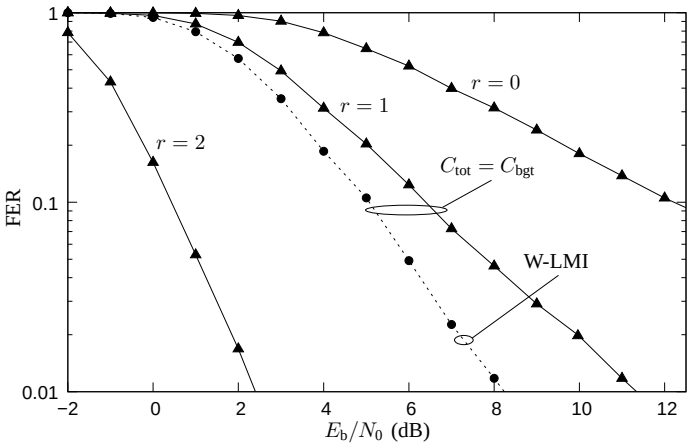


Fig. 8: Performance comparison. The same setting as in Fig. 5(d) is used except that the number of antennas is $N_{\text{R}} = N_{\text{T}} = 12$ and the coherence time is $L = 150$ ($K = 6$). The description of the curves is the same as in Fig. 5.

$\boldsymbol{l}(\boldsymbol{y})$ for one specific channel matrix $\boldsymbol{H}$. The PM method contains 2^{r+1} summation terms per detected bit and in each summation term in (6), the ZF method is performed. The ZF evaluations yield approximate solutions $\hat{\tilde{\boldsymbol{s}}} \in \mathcal{S}^{N_{\text{T}}-r-1}$ to the maximizations in (6) by using the model $\tilde{\boldsymbol{y}} = \tilde{\boldsymbol{H}}\tilde{\boldsymbol{s}} + \boldsymbol{e}$ where $\tilde{\boldsymbol{y}} \triangleq \boldsymbol{y} - \bar{\boldsymbol{H}}\bar{\boldsymbol{s}}$. To detect the bits transmitted through a specific $\boldsymbol{H}$, the PM method uses $N_{\text{T}} - r$ different partitions of $\boldsymbol{H} = [\bar{\boldsymbol{H}}\ \tilde{\boldsymbol{H}}]$ and $\boldsymbol{s} = [\bar{\boldsymbol{s}}^T\ \tilde{\boldsymbol{s}}^T]^T$. Since we have assumed block fading, some preprocessing of each channel matrix can be performed. Thus, the complexity cost can be divided into two parts, an initialization cost and a completion (evaluation) cost. Note that the completion stage, which is the heavy part, is performed L times more often than the initialization stage.

Initialization: Let $\gamma_{\text{init}}(r)$ contain the sum of operations in this stage and consider complexity counts of matrix manipulations performed flat-out. The bits permutation in PM is performed once per matrix $\boldsymbol{H}$ and requires roughly ${N_{\text{T}}}^3(r+1)$ FLOPs, which follows from $r+1$ matrix inversions of $\boldsymbol{H}^T\boldsymbol{H}$. Furthermore, for each partitioning of $\boldsymbol{H}$, PM performs:

Operation	**FLOPs**
QR-decomp. of $\tilde{\boldsymbol{H}} = \tilde{\boldsymbol{Q}}\tilde{\boldsymbol{R}}$, see [18]	$N_{\text{R}}(N_{\text{T}} - r - 1)^2$
eval. of $\tilde{\boldsymbol{Q}}^T\bar{\boldsymbol{H}}$	${N_{\text{R}}}^2(r+1)$
eval. of $\tilde{\boldsymbol{Q}}^T\bar{\boldsymbol{H}}\bar{\boldsymbol{s}}$	$2^{r+1}N_{\text{R}}(r+1)$

Completion: Let $\gamma_{\text{compl}}(r)$ contain the sum of operations per $\boldsymbol{y}$ vector in this stage. For each partitioning of $\boldsymbol{H}$, there are 2^{r+1} different $\tilde{\boldsymbol{y}}_{\text{q}} \triangleq \tilde{\boldsymbol{Q}}^T\tilde{\boldsymbol{y}} = \tilde{\boldsymbol{Q}}^T\boldsymbol{y} - \tilde{\boldsymbol{Q}}^T\bar{\boldsymbol{H}}\bar{\boldsymbol{s}}$ vectors and exponential terms $\exp\left(-\frac{1}{N_0}\left\|\tilde{\boldsymbol{y}}_{\text{q}} - \tilde{\boldsymbol{R}}\hat{\tilde{\boldsymbol{s}}}\right\|^2\right)$ in (6). Hence, for each such partitioning, PM performs:

Operation	**FLOPs**
eval. of $\tilde{\boldsymbol{Q}}^T\boldsymbol{y}$	${N_{\text{R}}}^2$
eval. of $\tilde{\boldsymbol{y}}_{\text{q}}$	$2^{r+1}N_{\text{R}}$
eval. of $\hat{\tilde{\boldsymbol{s}}}$ (backward subst.)	$2^{r+1}(N_{\text{T}} - r - 1)^2$
eval. of $\tilde{\boldsymbol{R}}\hat{\tilde{\boldsymbol{s}}}$	$2^{r+1}N_{\text{R}}(N_{\text{T}} - r - 1)$

Finally, for each detected bit, 2^{r+1} exponential terms are added. Since there are N_{T} bits, this requires in total $N_{\text{T}}2^{r+1}$

FLOPs.

Total: By considering only the largest terms, the total complexity of PM per matrix $\boldsymbol{H}$ when $L \gg N_{\mathrm{R}} \geq N_{\mathrm{T}}$ (slow fading) is

$$\begin{aligned}\gamma_{\text{tot}}(r) &\triangleq \gamma_{\text{init}}(r) + L\gamma_{\text{compl}}(r) \approx L\gamma_{\text{compl}}(r) \\ &= L(N_{\mathrm{T}} - r)\left[{N_{\mathrm{R}}}^2 + 2^{r+1}N_{\mathrm{R}}\right. \\ &\quad \left. + 2^{r+1}((N_{\mathrm{T}} - r - 1)^2 + N_{\mathrm{R}}(N_{\mathrm{T}} - r - 1) + N_{\mathrm{T}})\right] \\ &\approx 2^{r+1}L(N_{\mathrm{T}} - r)\left[(N_{\mathrm{T}} - r - 1)^2 + N_{\mathrm{R}}(N_{\mathrm{T}} - r - 1)\right] \\ &\quad + L(N_{\mathrm{T}} - r){N_{\mathrm{R}}}^2.\end{aligned}$$

When r is large, the 2^{r+1} term is dominating. In addition, for small r ($N_{\mathrm{T}} \gg r \geq 0$) and $N_{\mathrm{R}} \leq \frac{1+\sqrt{5}}{2}N_{\mathrm{T}}$, the polynomial next to the 2^{r+1} term is not smaller than $L(N_{\mathrm{T}} - r){N_{\mathrm{R}}}^2$. This means that the major complexity of PM with respect to r is mainly described by the 2^{r+1} expression, i.e., the rest can be considered constant.

ACKNOWLEDGMENTS

The simulations in this work were conducted with the help from the National Supercomputer Centre in Linköping Sweden (`www.nsc.liu.se`).

REFERENCES

[1] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. New York, NY, USA: Cambridge Uni. Press, 2005.
[2] E. Telatar, "Capacity of multi-antenna Gaussian channels," *European Transactions on Telecommunications*, vol. 10, pp. 585–596, Jun. 2001.
[3] E. G. Larsson, "MIMO detection methods: How they work," *IEEE Signal Processing Magazine*, vol. 26, pp. 91–95, May 2009.
[4] B. Hassibi and H. Vikalo, "On the sphere-decoding algorithm I. Expected complexity," *IEEE Transactions on Signal Processing*, vol. 53, pp. 2806–2818, Aug. 2005.
[5] L. G. Barbero and J. S. Thompson, "Fixing the complexity of the sphere decoder for MIMO detection," *IEEE Transactions on Wireless Communications*, vol. 7, pp. 2131–2142, Jun. 2008.
[6] E. G. Larsson and J. Jaldén, "Fixed-complexity soft MIMO detection via partial marginalization," *IEEE Transactions on Signal Processing*, vol. 56, pp. 3397–3407, Aug. 2008.
[7] D. Persson and E. G. Larsson, "Partial marginalization soft MIMO detection with higher order constellations," *IEEE Transactions on Signal Processing*, vol. 59, pp. 453–458, Jan. 2011.
[8] ITU-R Recommendation M.1225, "Guidelines for evaluation of radio transmission technologies for IMT-2000," 1997.
[9] D. W. Waters, N. Sommer, A. Batra, and S. Hosur, "Dynamic resource allocation to improve MIMO detection performance," U.S. Patent Application 0 137 762 A1, Jun. 12 2008.
[10] C. Studer, A. Burg, and H. Bölcskei, "Soft-output sphere decoding: algorithms and VLSI implementation," *IEEE Journal on Selected Areas in Communications*, vol. 26, pp. 290–300, Feb. 2008.
[11] I.-W. Lai, G. Ascheid, H. Meyr, and T. Chiueh, "Low-complexity channel-adaptive MIMO detection with just-acceptable error rate," in *Proc. IEEE 69th Vehicular Technology Conference (VTC)*, pp. 1–5, 2009.
[12] K. Nikitopoulos and G. Ascheid, "MIMO APP receiver processing with performance-determined complexity." http://arxiv.org/abs/1010.4160, Oct. 2010.
[13] M. Čirkić, D. Persson, and E. G. Larsson, "Optimization of computational resource allocation for soft MIMO detection," in *Proc. 43:rd Asilomar Conference on Signals, Systems and Computers*, pp. 1488–1492, 2009.
[14] M. Čirkić, D. Persson, and E. G. Larsson, "New results on adaptive computational resource allocation in soft MIMO detection," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2011.
[15] P. Fertl, J. Jaldén, and G. Matz, "Capacity-based performance comparison of MIMO-BICM demodulators," in *Proc. IEEE Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 166–170, 2008.
[16] J. Zhang, M. Armand, and P. Kam, "A mutual information approach for comparing LLR metrics for iterative decoders," in *Proc. IEEE International Conference on Communications (ICC)*, pp. 1–5, 2009.
[17] E. Denardo, *Dynamic Programming: Models and Applications*. New York, NY, USA: Dover Publications, 2003.
[18] G. Golub and C. Van Loan, *Matrix computations*. Baltimore, Maryland, USA: The Johns Hopkins Uni. Press, 1996.