Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2009, Article ID 645041, 13 pages
doi:10.1155/2009/645041

Research Article

Nonconcave Utility Maximisation in the MIMO Broadcast Channel

Johannes Brehmer and Wolfgang Utschick

Associate Institute for Signal Processing, Technische Universität München, 80333 Munich, Germany

Correspondence should be addressed to Johannes Brehmer, brehmer@tum.de

Received 15 February 2008; Accepted 12 June 2008

Recommended by S. Toumpis

The problem of determining an optimal parameter setup at the physical layer in a multiuser, multiantenna downlink is considered. An aggregate utility, which is assumed to depend on the users' rates, is used as performance metric. It is not assumed that the utility function is concave, allowing for more realistic utility models of applications with limited scalability. Due to the structure of the underlying capacity region, a two step approach is necessary. First, an optimal rate vector is determined. Second, the optimal parameter setup is derived from the optimal rate vector. Two methods for computing an optimal rate vector are proposed. First, based on the differential manifold structure offered by the boundary of the MIMO BC capacity region, a gradient projection method on the boundary is developed. Being a local algorithm, the method converges to a rate vector which is not guaranteed to be a globally optimal solution. Second, the monotonic structure of the rate space problem is exploited to compute a globally optimal rate vector with an outer approximation algorithm. While the second method yields the global optimum, the first method is shown to provide an attractive tradeoff between utility performance and computational complexity.

1. Introduction

The majority of current wireless communication systems are based on the principle of *orthogonal multiple access*. Simply speaking, multiple users compete for a set of shared channels, and access to the channels is coordinated such that each channel is used by a single user only. The decision which user accesses which channel is made at the *medium access* (MAC) layer, with the result that at the *physical* (PHY) layer, transmission is over single-user channels. Based on recent advances in physical layer techniques such as MIMO signal processing and multiuser coding, it has been shown that significant performance gains can be achieved by allowing one channel to be used by multiple users at once [1–5]. In other words, the physical layer paradigm is shifting from single-user channels to multiuser channels. This change also dissolves the strict distinction between MAC and PHY layers, as the question which users access which channels can only be answered in a joint treatment of both layers.

In this work, a multiuser, multiantenna downlink in a single-cell wireless system is considered, which, from the viewpoint of information theory, corresponds to a *MIMO broadcast channel* (MIMO BC) [3, 6]. While the aforementioned shift to multiuser channels is motivated by the potential gains in system performance, an evident drawback of this shift is the increased design complexity. In other words, multiantenna, multiuser channels significantly increase the set of design parameters and degrees of freedom at the PHY layer. Clearly, strategies for tuning these parameters in an optimal manner are of great interest.

The desire for maximum system performance leads immediately to the question of optimality criteria. While voice and best effort data applications have been predominant, future wireless systems are expected to provide a multitude of heterogeneous applications, ranging from best effort data to low-delay gaming applications, from low-rate messaging to high-rate video. The heterogeneity of these applications requires application-aware optimality criteria, that is, it is no longer sufficient to optimise PHY and MAC layers with respect to criteria such as average throughput or proportional rate fairness. Utility functions have been widely used as a model for the properties of upper layers.

In this work, the focus is on the optimisation of the PHY layer parameters, and a generic utility model in terms of a function that is monotone in the users' rates is employed. For a wide range of applications, utility models can be found in the literature. In [7], applications are classified based on their elasticity with respect to the allocated rate. Best effort applications can be modelled with a concave utility [7]. On the other hand, less elastic applications result in a nonconcave utility model [7, 8]. While most works on utility maximisation in wireless systems assume concave utilities, the nonconcave setup has received relatively little attention [8–10]. Based on the premise that some relevant application classes can be more precisely modelled by nonconcave utilities, this work proposes a solution strategy that provides at least locally optimal performance in the nonconcave case.

There exists a significant amount of literature on utility maximisation for wireless networks, see, for example, [10–13] and references therein. The network-oriented works usually consider a large number of nodes with a simple physical layer setup, and focus on computationally efficient and distributed resource allocation strategies for large networks. In contrast, this work focuses on the optimisation of a limited-size infrastructure network with a complex multiantenna, multiuser PHY/MAC layer configuration. Utility maximisation in the MIMO BC is also investigated in [14]. The authors solve the utility maximisation problem based on Lagrange duality, under the assumption of concave utility functions. *Dual methods* are frequently used in network utility maximisation [10], but rely on the assumption of problem convexity. This work makes the following contributions. First, a primal gradient-based method for addressing the utility maximisation problem in the MIMO BC is developed. The proposed method does not rely on a convexity assumption and can provide convergence to local optima in the nonconvex case. The quality of such local solutions depends on the specific problem instance and can only be evaluated if the global optimum is known. The second contribution of this work is the application of methods from the field of deterministic global optimisation to the nonconcave utility maximisation problem. It is shown that the utility maximisation problem in the MIMO BC can be cast as a monotonic optimisation problem [15]. The monotonicity structure can be exploited to efficiently find the global optimum by an outer approximation algorithm.

Notation. Vectors and vector-valued functions are denoted by bold lowercase letters, matrices by bold uppercase letters. The transpose and the Hermitian transpose of $\mathbf{Q}$ are denoted by $\mathbf{Q}^{\mathrm{T}}$ and $\mathbf{Q}^{\mathrm{H}}$, respectively. The identity matrix is denoted by $\mathbf{1}$. Concerning boldface, an exception is made for gradients. The gradient of a function u evaluated at $\mathbf{x}$ is a vector $\nabla u(\mathbf{x})$, the gradient of a function $\mathbf{f}$ evaluated at $\mathbf{x}$ is a matrix $\nabla \mathbf{f}(\mathbf{x})$ whose ith column is the gradient at x of the ith component function of $\mathbf{f}$ [16]. The following definitions of order relations between vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^K$, with $K > 1$, are used:

$$\begin{aligned} \mathbf{x} \geq \mathbf{y} &\iff \forall k : x_k \geq y_k, \\ \mathbf{x} > \mathbf{y} &\iff \mathbf{x} \geq \mathbf{y}, \exists k : x_k > y_k, \\ \mathbf{x} \gg \mathbf{y} &\iff \forall k : x_k > y_k. \end{aligned} \tag{1}$$

Order relations $\leq, <, \ll$ are defined in the same manner.

2. Problem Setup

At the physical layer, a MIMO broadcast channel with K receivers is considered. The transmitter has N transmit antennas, while receiver k is equipped with M_k receiving antennas. The transmitter sends independent information to each of the receivers.

The received signal at receiver k is given by

$$\mathbf{y}_k = \mathbf{H}_k \sum_{i=1}^{K} \mathbf{x}_i + \boldsymbol{\eta}_k, \tag{2}$$

where $\mathbf{H}_k \in \mathbb{C}^{M_k \times N}$ is the channel to receiver k and $\mathbf{x}_k \in \mathbb{C}^N$ is the signal transmitted to receiver k. Furthermore, $\boldsymbol{\eta}_k$ is the circularly symmetric complex Gaussian noise at receiver k, with $\boldsymbol{\eta}_k \sim \mathcal{CN}(\mathbf{0}, \mathbf{1}_{M_k})$.

Let $\mathbf{Q}_k$ denote the transmit covariance matrix of user k. The total transmit power has to satisfy the power constraint $\mathrm{tr}(\sum_{k=1}^{K} \mathbf{Q}_k) \leq P_{\mathrm{tr}}$. Accordingly, with $\mathbf{Q} = (\mathbf{Q}_1, \ldots, \mathbf{Q}_K)$ the set of feasible transmit covariance matrices is given by

$$\mathcal{Q} = \left\{ \mathbf{Q} : \mathbf{Q}_k \in \mathbb{H}_+^N, \mathrm{tr}\left(\sum_{k=1}^{K} \mathbf{Q}_k \right) \leq P_{\mathrm{tr}} \right\}, \tag{3}$$

where $\mathbb{H}_+^N$ denotes the set of positive semidefinite Hermitian $N \times N$ matrices.

As proved in [6], capacity is achieved by *dirty paper coding* (DPC). Let π denote the encoding order, that is, $\pi : \{1, \ldots, K\} \to \{1, \ldots, K\}$ is a permutation, and $\pi(i)$ is the index of the user which is encoded at the ith position. Moreover, let Π denote the set of all possible permutations on $\{1, \ldots, K\}$.

For fixed $\mathbf{Q}$ and π, an achievable rate vector is given by $\mathbf{r}(\mathbf{Q}, \pi) = (r_1(\mathbf{Q}, \pi), \ldots, r_K(\mathbf{Q}, \pi))$, with

$$r_{\pi(i)} = \log \frac{\det\left(\mathbf{1} + \mathbf{H}_{\pi(i)} \left(\sum_{j \geq i} \mathbf{Q}_{\pi(j)} \right) \mathbf{H}_{\pi(i)}^{\mathrm{H}} \right)}{\det\left(\mathbf{1} + \mathbf{H}_{\pi(i)} \left(\sum_{j > i} \mathbf{Q}_{\pi(j)} \right) \mathbf{H}_{\pi(i)}^{\mathrm{H}} \right)}. \tag{4}$$

Let $\mathcal{R}$ denote the set of rate vectors achievable by feasible $\mathbf{Q}$ and π:

$$\mathcal{R} = \{ \mathbf{r}(\mathbf{Q}, \pi) : \mathbf{Q} \in \mathcal{Q}, \pi \in \Pi \}. \tag{5}$$

The capacity region of the MIMO BC is defined as the convex hull of $\mathcal{R}$ [3]:

$$\mathcal{C} = \mathrm{co}(\mathcal{R}). \tag{6}$$

Accordingly, each element of $\mathcal{C}$ can be written as a convex combination of elements of $\mathcal{R}$, that is, for each $\mathbf{r} \in \mathcal{C}$, there exists a set of coefficients $\{\alpha_w\}$, a set of transmit covariance matrices $\{\mathbf{Q}^{(w)}\}$, and a set of encoding orders $\{\pi^{(w)}\}$ such that

$$\mathbf{r} = \sum_{w=1}^{W} \alpha_w \mathbf{r}(\mathbf{Q}^{(w)}, \pi^{(w)}), \tag{7}$$

with $\alpha_w \geq 0$, $\sum_{w=1}^{W} \alpha_w = 1$, $\mathbf{Q}^{(w)} \in \mathcal{Q}$, and $\pi^{(w)} \in \Pi$. In other words, $\mathbf{r}$ is achieved by time-sharing between rate vectors $\mathbf{r}(\mathbf{Q}^{(w)}, \pi^{(w)}) \in \mathcal{R}$.

Each $\mathbf{r} \in \mathcal{C}$ can be achieved by time-sharing between at most K rate vectors $\mathbf{r}(\mathbf{Q}^{(w)}, \pi^{(w)}) \in \mathcal{R}$, thus $W \leq K$. Accordingly, the physical layer parameter vector can be defined as follows:

$$\mathbf{x}_{\mathrm{P}} = \left(\alpha_w, \mathbf{Q}^{(w)}, \pi^{(w)}\right)_{w=1}^{K}. \tag{8}$$

Moreover, the set of feasible PHY parameter setups is given by

$$\mathcal{X}_{\mathrm{P}} = \left\{ \mathbf{x}_{\mathrm{P}} : \alpha_w \geq 0, \sum_{w=1}^{W} \alpha_w = 1, \mathbf{Q}^{(w)} \in \mathcal{Q}, \pi^{(w)} \in \Pi \right\}. \tag{9}$$

Given the set $\mathcal{X}_{\mathrm{P}}$, an obvious problem is finding a parameter setup $\mathbf{x}_{\mathrm{P}}^{*}$, that is, in a desired sense, optimal.

In this work, it is assumed that the properties of the upper layers are summarised in a system utility function $u : \mathbb{R}_{+}^{K} \to \mathbb{R}$, whose value depends only on the rate vector provided by the physical layer. The parameter optimisation problem is then given by

$$\max_{\mathbf{x}_{\mathrm{P}}} u\left(\mathbf{r}(\mathbf{x}_{\mathrm{P}})\right) \quad \text{s.t. } \mathbf{x}_{\mathrm{P}} \in \mathcal{X}_{\mathrm{P}}, \tag{10}$$

where $\mathbf{r}(\mathbf{x}_{\mathrm{P}})$ follows from (7). Concerning the function u, it is assumed that larger rates result in higher utility, that is, it is assumed that u is strictly monotonically increasing. Strict monotonicity implies that

$$\mathbf{r} > \mathbf{r}' \Longrightarrow u(\mathbf{r}) > u(\mathbf{r}'). \tag{11}$$

Moreover, it is assumed that u is continuous, and differentiable on $\mathbb{R}_{++}^{K}$. The function u is not assumed to be concave.

3. Nonconcave Utilities

One of the premises of this work is that nonconcave utilities are of high practical relevance in future communication systems. Consider the case $K = 1$. A strictly monotone function $u : r \mapsto u(r)$ is concave if the gain in utility obtained from increasing r decreases with increasing r, for all $r \in \mathbb{R}_{+}$. A common example for such a behaviour is best effort data applications, where any increase in rate is good, but a saturation effect leads to a decreasing gain for larger r [7]. Such elastic applications are perfectly scalable. On the other extreme, applications that have fixed rate requirements (such as traditional voice service) are not scalable at all (inelastic) and are more precisely modelled by a nonconcave utility. Below a certain rate threshold, utility is zero, above the threshold utility takes on its maximum value (step function) [7].

Based on recent advances in multimedia coding, future multimedia applications can be expected to lie between these two extremes. They are scalable to some extent, but do not provide the perfect scalability of best effort services. As an example, the scalable video coding extension of the H.264/AVC standard [17] provides support of scalability based on a layered video codec. Due to the finite number of layers, the decoded video's quality only increases at those rates where an additional layer can be transmitted. Moreover, if the gain between layers is not incremental (such as experienced when switching between low and high spatial resolution), such a behaviour can be more precisely modelled by a nonconcave utility, which, in contrast to a concave utility, does not require a steady decrease of the gain over the whole range of feasible rates. To summarise, the flexibility offered by nonconcave utilities allows for more precise models of multimedia applications, which only have a finite number of operation modes and show a nonmonotone behaviour of the gains experienced by an increase in rate.

4. Direct Approach

Based on (10), a first approach may be to directly optimise the composite function $u \circ r$ with respect to the PHY parameters $\mathbf{x}_{\mathrm{P}}$. In general, however, this approach will fail, due to the discrete nature of Π and the nonconvexity of problem (10), even for a concave utility function u.

In contrast, the capacity region is convex by definition, thus the problem

$$\max_{\mathbf{r}} u(\mathbf{r}) \quad \text{s.t. } \mathbf{r} \in \mathcal{C} \tag{12}$$

is convex for concave u. This motivates solution approaches that operate in the rate space and not in the physical layer parameter space.

A special case for which the direct approach succeeds is given by the utility $u(\mathbf{r}) = \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{r}$, that is, *weighted sum rate maximisation* (WsrMax). In this case, time sharing is not required, that is, $\alpha_w^{*} = 0$, $w > 1$. Moreover, the gradient ∇u is independent of $\mathbf{r}$, and an optimal encoding order π^{*} can be directly inferred from $\boldsymbol{\lambda}$ [3, 4, 18]. As a result, the problem is reduced to find the optimal transmit covariance matrices, which can be solved as a convex problem in the dual MAC [4]. Denote by $\mathbf{r}_{\mathrm{wsr}}(\boldsymbol{\lambda}, \pi^{*})$ the rate vector that maximises weighted sum rate for a given weight $\boldsymbol{\lambda}$ and a corresponding optimal encoding order π^{*}, that is,

$$\boldsymbol{\lambda}^{\mathrm{T}}\mathbf{r}_{\mathrm{wsr}}(\boldsymbol{\lambda}, \pi^{*}) = \max_{\mathbf{Q} \in \mathcal{Q}} \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{r}(\mathbf{Q}, \pi^{*}). \tag{13}$$

For general utility functions, the optimal solution may require time-sharing. In particular, if no further assumptions concerning the properties of u are made, the loss incurred by approximating a time-sharing solution by a rate vector $\mathbf{r} \in \mathcal{R}$ may be significant. Moreover, even if the optimal solution does not require time-sharing, it is not clear how to find the optimal encoding order.

An optimisation algorithm operating in the rate space of course still requires a means to compute points from $\mathcal{C}$. WsrMax over $\mathcal{C}$ can be cast as a convex problem. Moreover, efficient algorithms for solving the WsrMax problem in the MIMO BC have been proposed recently [19, 20]. Based on this observation, the proposed algorithm is formulated such that iterates on $\mathcal{C}$ are obtained as solutions of WsrMax problems.

5. Iterative Efficient Set Approximation

To solve problem (10), a two-step procedure is followed. First, determine a (possibly locally) optimal solution $\mathbf{r}^*$ of problem (12) by operating in the rate space. Second, given $\mathbf{r}^*$, determine a parameter setup $\mathbf{x}_{\mathrm{p}}^*$ such that

$$\mathbf{r}(\mathbf{x}_{\mathrm{p}}^*) = \mathbf{r}^*. \tag{14}$$

Due to the assumed strict monotonicity of the function u, all candidate solutions to problem (10) lie on the Pareto efficient boundary of $\mathcal{C}$. The Pareto efficient set is defined as

$$\mathcal{E} = \{\mathbf{r} \in \mathcal{C} : \nexists \mathbf{r}' \in \mathcal{C} : \mathbf{r}' > \mathbf{r}\}. \tag{15}$$

Knowing that $\mathbf{r}^* \in \mathcal{E}$, a gradient projection method is proposed that generates iterates on $\mathcal{E}$. Note that there exist different flavours of gradient projection methods, a gradient projection on arbitrary convex sets [16], requiring a Euclidean projection and a gradient projection on sets, equipped with a differential manifold structure [21–23]. In this work, the second approach is followed.

In the classical gradient projection method of Rosen [24], it is assumed that the feasible set is described by a set of constraint functions $\mathbf{h}$, $\mathbf{m}$ such that the set of feasible $\mathbf{r}$ is given by $\mathbf{h}(\mathbf{r}) \leq \mathbf{0}$, $\mathbf{m}(\mathbf{r}) = \mathbf{0}$ with $\mathbf{h}$, $\mathbf{m}$ differentiable. For the capacity region of the MIMO BC, such a description in terms of constraint functions in $\mathbf{r}$ is not available (basically, all that is available is a method to compute points on its efficient boundary, by means of WsrMax). The key for a gradient-based optimisation in the rate space is to recognise the differentiable manifold structure offered by the efficient boundary of the capacity region. By exploiting this structure, a gradient ascent on $\mathcal{E}$ that does not rely on a description in terms of constraint functions is possible.

5.1. Gradient Ascent on $\mathcal{E}$. The following problem is considered:

$$\max_{\mathbf{r} \in \mathcal{E}} u(\mathbf{r}). \tag{16}$$

The efficient set $\mathcal{E}$ is a $K-1$ dimensional manifold with boundary [25], where the boundary of $\mathcal{E}$ corresponds to rate vectors $\mathbf{r} \in \mathcal{E}$ with at least one user having zero rate. Furthermore, it is assumed that for the MIMO BC, the interior of the efficient set, defined by

$$\widetilde{\mathcal{E}} = \{\mathbf{r} \in \mathcal{E} : \mathbf{r} \gg \mathbf{0}\}, \tag{17}$$

is smooth up to first order, that is, $\widetilde{\mathcal{E}}$ is a C^1 differentiable [25], $K-1$ dimensional manifold. Based on this assumption, there exists a set $\{\boldsymbol{\phi}_{\mathbf{r}}\}_{\mathbf{r} \in \widetilde{\mathcal{E}}}$ of differentiable local parameterisations $\boldsymbol{\phi}_{\mathbf{r}} : \mathcal{U}_{\mathbf{r}} \subset \mathbb{R}^{K-1} \to \widetilde{\mathcal{E}}$, with $\mathcal{U}_{\mathbf{r}}$ open and $\boldsymbol{\phi}_{\mathbf{r}}(\mathbf{0}) = \mathbf{r}$ [25].

For simplicity, it is first assumed that $\mathbf{r}^* \in \widetilde{\mathcal{E}}$. Based on this assumption, starting at $\mathbf{r}^{(0)}$, a sequence of iterates $\mathbf{r}^{(n)} \in \widetilde{\mathcal{E}}$ is generated. At each $\mathbf{r}^{(n)}$, a parameterisation $\boldsymbol{\phi}_{\mathbf{r}^{(n)}}$ is available. Composing parameterisation and utility function results in a function $f_{\mathbf{r}} = u \circ \boldsymbol{\phi}_{\mathbf{r}}$, which maps an open subset of $\mathbb{R}^{K-1}$ into $\mathbb{R}$. The composite function $f_{\mathbf{r}}$ is amenable to standard methods for unconstrained optimisation. Based on this observation, a gradient ascent is carried out on the set of functions $f_{\mathbf{r}} = u \circ \boldsymbol{\phi}_r$. Let $\mathbf{r}^{(n)}$ denote the nth iterate, and let $\boldsymbol{\mu}^{(n)}$ denote its coordinates in the parameterisation $\boldsymbol{\phi}_{\mathbf{r}^{(n)}}$, that is, $\boldsymbol{\mu}^{(n)} = \boldsymbol{\phi}_{\mathbf{r}^{(n)}}^{-1}(\mathbf{r}^{(n)}) = \mathbf{0}$. By definition of $f_{\mathbf{r}}$, $u(\mathbf{r}) = f_{\mathbf{r}}(\mathbf{0})$. The composite function $f_{\mathbf{r}}$ is differentiable at $\mathbf{0}$, with gradient $\nabla f_{\mathbf{r}}$ at $\mathbf{0}$ given by

$$\nabla f_{\mathbf{r}}(\mathbf{0}) = \nabla \boldsymbol{\phi}_{\mathbf{r}}(\mathbf{0}) \nabla u(\mathbf{r}), \tag{18}$$

where $\nabla \boldsymbol{\phi}_{\mathbf{r}}^{\mathrm{T}}$ is the Jacobian of $\boldsymbol{\phi}_{\mathbf{r}}$. If $\nabla f_{\mathbf{r}}(\mathbf{0}) \neq \mathbf{0}$, then $\nabla f_{\mathbf{r}}(\mathbf{0})$ is an ascent direction of $f_{\mathbf{r}}$ at $\mathbf{0}$, that is, there exists a $\beta > 0$ such that for all t, $0 < t \leq \beta$,

$$t \nabla f_{\mathbf{r}}(\mathbf{0}) \in \mathcal{U}_{\mathbf{r}}, \tag{19}$$

$$f_{\mathbf{r}}(t \nabla f_{\mathbf{r}}(\mathbf{0})) > f_{\mathbf{r}}(\mathbf{0}), \tag{20}$$

where (19) follows from the fact that $\mathcal{U}_{\mathbf{r}}$ is open and (20) from the differentiability of $f_{\mathbf{r}}$, see, for example, [26, Theorem 2.1]. This gives rise to the following iteration:

$$\boldsymbol{\mu}^{(n)} = \boldsymbol{\phi}_{\mathbf{r}^{(n)}}^{-1}(\mathbf{r}^{(n)}) = \mathbf{0}, \tag{21}$$

$$\boldsymbol{\mu}^{(n+1)} = t \nabla f_{\mathbf{r}^{(n)}}(\mathbf{0}), \tag{22}$$

$$\mathbf{r}^{(n+1)} = \boldsymbol{\phi}_{\mathbf{r}^{(n)}}(\boldsymbol{\mu}^{(n+1)}), \tag{23}$$

with $t > 0$ chosen such that properties (19) and (20) are fulfilled. The algorithm defined in (21)–(23) is a so-called *varying parameterisation approach* to optimisation on manifolds [23, 27].

According to (20), the iterates $\mathbf{r}^{(n)}$ generate an increasing sequence $u(\mathbf{r}^{(n)})$. The iteration stops if

$$\nabla f_{\mathbf{r}}(\mathbf{0}) = \mathbf{0}. \tag{24}$$

In this work, points $\mathbf{r} \in \mathcal{E}$ for which (24) holds are denoted as *stationary points*. The tangent space of $\widetilde{\mathcal{E}}$ at $\mathbf{r}$ is defined as

$$\mathcal{T}_{\mathbf{r}} = \operatorname{span}(\nabla \boldsymbol{\phi}_{\mathbf{r}}(\mathbf{0})^{\mathrm{T}}). \tag{25}$$

Thus, geometrically, stationary points correspond to points on the efficient boundary where the gradient of the utility function is orthogonal to the tangent space (cf. (18)). In the context of minimising a differentiable function over a differentiable manifold, (24) represents a necessary first-order optimality condition [22].

The step size t is determined with an inexact line search. As evaluations of $f_{\mathbf{r}}$ are usually computationally expensive, the step size t is chosen such that an increase in the utility value results, while keeping the number of evaluations of $f_{\mathbf{r}}$ as small as possible. Define

$$\theta(t) = f_{\mathbf{r}}(t \nabla f_{\mathbf{r}}(\mathbf{0})) = u(\boldsymbol{\phi}_{\mathbf{r}^{(n)}}(t \nabla f_{\mathbf{r}^{(n)}}(\mathbf{0}))). \tag{26}$$

Starting with an initial step size $t = t_0$ that satisfies (19), the step size t is halved until

$$\theta(t) \geq \theta(0) + \alpha \nabla \theta(0) t, \tag{27}$$

for fixed α, $0 < \alpha < 1$. Note that (27) corresponds to Armijo's rule [28] for accepting a step size as not too large. In contrast

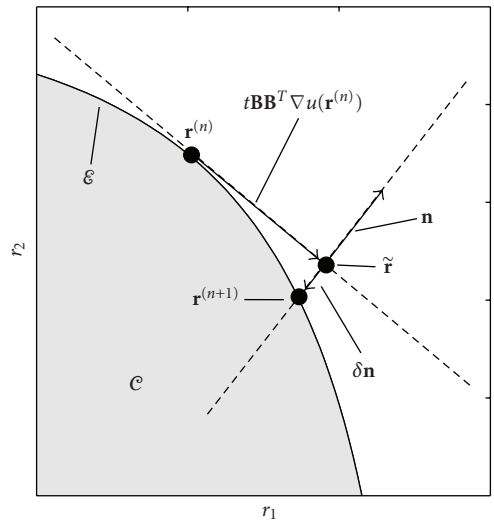

Figure 1: One iteration of the IEA method.

to Armijo's rule, however, there is no test whether the step size is too small, that is, t_0 is always considered large enough.

There exists a choice for the parameterisations $\boldsymbol{\phi}_{\mathbf{r}}$ for which $\nabla\boldsymbol{\phi}_{\mathbf{r}}(\mathbf{0})$, and thus $\nabla f_{\mathbf{r}}(\mathbf{0})$, is particularly simple to compute. Let $\mathbf{B} \in \mathbb{R}^{K\times K-1}$ denote an orthonormal basis of the tangent space $\mathcal{T}_{\mathbf{r}}$. Choose $\mathbf{n}$ such that the columns of $[\mathbf{B}\ \ \mathbf{n}]$ constitute an orthonormal basis of $\mathbb{R}^K$. Choose the parameterisation $\boldsymbol{\phi}_{\mathbf{r}}$ as follows:

$$\boldsymbol{\phi}_{\mathbf{r}}(\boldsymbol{\mu}) = \mathbf{r} + \mathbf{B}\boldsymbol{\mu} + \mathbf{n}\delta(\boldsymbol{\mu}), \tag{28}$$

where $\delta(\boldsymbol{\mu})$ is chosen such that $\boldsymbol{\phi}_{\mathbf{r}}(\boldsymbol{\mu}) \in \widetilde{\mathcal{E}}$ (correction step). Then

$$\nabla\boldsymbol{\phi}_{\mathbf{r}}(\mathbf{0}) = \mathbf{B}^{\mathrm{T}}. \tag{29}$$

As shown in Section 5.2, it is straightforward to find a basis $\mathbf{B}$. Combining (22), (23), (18), (28), and (29) yields

$$\mathbf{r}^{(n+1)} = \mathbf{r}^{(n)} + t\mathbf{B}\mathbf{B}^{\mathrm{T}}\nabla u\big(\mathbf{r}^{(n)}\big) + \mathbf{n}\delta(t), \tag{30}$$

with $\delta(t) = \delta(t\mathbf{B}^{\mathrm{T}}\nabla u(\mathbf{r}^{(n)})$. Accordingly, the update in rate space is given by

$$\mathbf{r}^{(n+1)} - \mathbf{r}^{(n)} = t\mathbf{B}\mathbf{B}^{\mathrm{T}}\nabla u\big(\mathbf{r}^{(n)}\big) + \mathbf{n}\delta(t). \tag{31}$$

The first summand in (31) is the orthogonal projection of $\nabla u(\mathbf{r}^{(n)})$ on the tangent space. Based on this observation, the proposed method can be interpreted as follows. First, approximate the efficient set by its tangent space at $\mathbf{r}^{(n)}$. Next, compute a gradient step, using this approximation. Finally, make a correction step from the approximation back to the efficient set, yielding $\mathbf{r}^{(n+1)}$. Based on the observation that at each iteration, an approximation of the efficient set is computed, the proposed method is denoted as *iterative efficient set approximation* (IEA). For the case of $K = 2$ users, one iteration of the IEA method is illustrated in Figure 1.

Equation (19) defines an upper bound on the step size t, which ensures that $\boldsymbol{\mu}^{(n+1)}$ stays within the domain of the parameterisation $\boldsymbol{\phi}_{\mathbf{r}^{(n)}}$. The domain of the parameterisation defined in (28) is defined implicitly by the requirement that all entries of the resulting rate vector have to be positive, that is,

$$\mathcal{U}_{\mathbf{r}} = \{\boldsymbol{\mu} : \boldsymbol{\phi}_{\mathbf{r}}(\boldsymbol{\mu}) \gg \mathbf{0}\}. \tag{32}$$

In fact, the image and domain of the parameterisation defined in (28) can be extended to also include rate vectors with zero entries. From (32) and (30), an upper bound on the step size t can then be derived by interpreting $\mathbf{r}^{(n+1)}$ as a function of t. An upper bound on t is given by the value of t where the smallest entry in $\mathbf{r}^{(n+1)}(t)$ is exactly zero:

$$\bar{t} : \min_k \mathbf{r}_k^{(n+1)}(\bar{t}) = 0. \tag{33}$$

Note that by (30), the upper bound $\bar{t}$ depends on $\mathbf{r}^{(n)}$—thus the validity range $0 < t < \bar{t}$ changes over $\widetilde{\mathcal{E}}$, and it may get small close to the boundary of $\mathcal{E}$.

5.2. Correction Step. The most involved step is the computation of $\delta(\boldsymbol{\mu}^{(n+1)})$. Write $\mathbf{r}^{(n+1)}$ as

$$\mathbf{r}^{(n+1)} = \tilde{\mathbf{r}} + \delta\mathbf{n}, \tag{34}$$

with $\tilde{\mathbf{r}} = \mathbf{r}^{(n)} + \mathbf{B}\boldsymbol{\mu}^{(n+1)}$. Based on (34), the correction step can be interpreted as the projection of $\tilde{\mathbf{r}}$ on $\mathcal{E}$ by computing the intersection between $\mathcal{E}$ and the line $\{\mathbf{r} = \tilde{\mathbf{r}} + x\mathbf{n}, x \in \mathbb{R}\}$, compare Figure 1. Assume that $\mathbf{n} \geq \mathbf{0}$ (the validity of this assumption is verified at the end of this subsection). Then, δ can be found by solving the following optimisation problem:

$$\delta = \max_{x,\mathbf{r}} x \quad \text{s.t. } \tilde{\mathbf{r}} + x\mathbf{n} \leq \mathbf{r},\ \mathbf{r} \in \mathcal{C}. \tag{35}$$

Note that (35) is a convex problem. In particular, it is independent of the utility function u, that is, it is convex regardless whether u is concave or not. Moreover, Slater's condition is satisfied, that is, strong duality holds. Accordingly, (35) can be solved via Lagrange duality.

The Lagrangian of problem (35) is given by

$$L(x, \mathbf{r}, \boldsymbol{\lambda}) = x + \boldsymbol{\lambda}^{\mathrm{T}}(\mathbf{r} - \tilde{\mathbf{r}} - x\mathbf{n}). \tag{36}$$

The dual function follows as

$$\begin{aligned} g(\boldsymbol{\lambda}) &= \sup_{\substack{x\in\mathbb{R}\\ \mathbf{r}\in\mathcal{C}}} \big(x(1 - \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{n}) + \boldsymbol{\lambda}^{\mathrm{T}}(\mathbf{r} - \tilde{\mathbf{r}})\big) \\ &= \begin{cases} +\infty, & \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{n} \neq 1, \\ \max\limits_{\mathbf{r}\in\mathcal{C}} \boldsymbol{\lambda}^{\mathrm{T}}(\mathbf{r} - \tilde{\mathbf{r}}), & \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{n} = 1. \end{cases} \end{aligned} \tag{37}$$

Note that for $\boldsymbol{\lambda}^{\mathrm{T}}\mathbf{n} = 1$, again a weighted sum-rate maximisation problem is to be solved. Recall from Section 4 that WsrMax can be efficiently solved as a convex problem in the dual MAC.

Let $\mathbf{r}^*(\boldsymbol{\lambda})$ denote a maximiser of the weighted sum-rate maximisation in (37) for a given $\boldsymbol{\lambda} \in \mathbb{R}_+^K$. The optimal dual variable $\boldsymbol{\lambda}$ is found by solving

$$\min_{\boldsymbol{\lambda}\geq\mathbf{0}} \boldsymbol{\lambda}^{\mathrm{T}}(\mathbf{r}^*(\boldsymbol{\lambda}) - \tilde{\mathbf{r}}) \quad \text{s.t. } \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{n} = 1. \tag{38}$$

According to Danskin's Theorem [16], a subgradient (at $\boldsymbol{\lambda}$) of the cost function of problem (38) is given by $(\mathbf{r}^*(\boldsymbol{\lambda}) - \widetilde{\mathbf{r}})$. If $\boldsymbol{\lambda}$ has equal entries, $\mathbf{r}^*(\boldsymbol{\lambda})$ is not unique [4]. Thus, the subgradient is not unique, and the cost function is nondifferentiable. Accordingly, the minimisation in (38) has to be carried out using any of the methods for nondifferentiable convex optimisation, such as subgradient methods, cutting plane methods, or the ellipsoid method [29]. All these methods have in common that they generate iterates $\boldsymbol{\lambda}^{(i)}$ (which converge to the optimal dual variable $\boldsymbol{\lambda}^*$), and at each iteration i, they require the computation of a subgradient at $\boldsymbol{\lambda}^{(i)}$—which basically corresponds to solving a WsrMax problem with weight $\boldsymbol{\lambda}^{(i)}$. In this work, an outer-linearisation cutting plane method [16] is used to solve problem (38).

As strong duality holds, $\delta = g(\boldsymbol{\lambda}^*)$, and

$$\mathbf{r}^{(n+1)} = \widetilde{\mathbf{r}} + g(\boldsymbol{\lambda}^*)\mathbf{n}. \tag{39}$$

From the optimal dual variable $\boldsymbol{\lambda}^*$ also follows the tangent space at $\mathbf{r}^{(n+1)}$. Due to strong duality, $\mathbf{r}^{(n+1)}$ maximises $L(x^*, \mathbf{r}, \boldsymbol{\lambda}^*)$ over $\mathcal{C}$ [16]. Accordingly, $\mathbf{r}^{(n+1)}$ is a maximiser of a WsrMax problem with weight $\boldsymbol{\lambda}^*$. Recall that for WsrMax, $u(\mathbf{r}) = \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{r}$, with $\nabla u(\mathbf{r}) = \boldsymbol{\lambda}$. The corresponding composite function $f_{\mathbf{r}}$ is given by $f_{\mathbf{r}}(\boldsymbol{\mu}) = \boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{\phi}_{\mathbf{r}}(\boldsymbol{\mu})$. As $\mathbf{r}^{(n+1)}$ is a maximiser of the WsrMax problem, it has to be a stationary point (for this particular composite function, with $\boldsymbol{\lambda} = \boldsymbol{\lambda}^*$). From (24), it follows that:

$$\nabla\left((\boldsymbol{\lambda}^*)^{\mathrm{T}}\boldsymbol{\phi}_{\mathbf{r}^{(n+1)}}\right)(\mathbf{0}) = \nabla\boldsymbol{\phi}_{\mathbf{r}^{(n+1)}}(\mathbf{0})\boldsymbol{\lambda}^* = \mathbf{0}, \tag{40}$$

thus

$$\mathcal{T}_{\mathbf{r}^{(n+1)}} = \operatorname{null}\left((\boldsymbol{\lambda}^*)^{\mathrm{T}}\right). \tag{41}$$

In other words, the basis $\mathbf{B}$ needed in the *next* iteration can be obtained by computing an orthonormal basis of the null space of $(\boldsymbol{\lambda}^*)^{\mathrm{T}}$, where $\boldsymbol{\lambda}^*$ is the optimal dual variable of the current iteration. In addition, in the next iteration a unit vector $\mathbf{n} \geq \mathbf{0}$ orthogonal to $\mathbf{B}$ is needed. From (41), it follows that $\mathbf{n}$ (in the next iteration) is simply

$$\mathbf{n} = \frac{\boldsymbol{\lambda}^*}{||\boldsymbol{\lambda}^*||_2}. \tag{42}$$

5.3. Time-Sharing Solutions. The algorithm described in Sections 5.1 and 5.2 yields a stationary point $\mathbf{r}^*$ of problem (12). The final step is the recovery of an optimal parameter setup $\mathbf{x}_{\mathrm{p}}^*$ from $\mathbf{r}^*$. The complexity of the recovery step depends on the location of $\mathbf{r}^*$. If $\mathbf{r}^* \notin \mathcal{R}$, then $\mathbf{r}^*$ lies in a time-sharing region. Throughout this work, the term *time-sharing region* denotes a subset of $\mathcal{E}$ whose elements are only achievable by time-sharing. In case of time-sharing optimality, the optimal parameter setup has to be found by identifying a set of points in $\mathcal{E} \cap \mathcal{R}$ whose convex combination yields $\mathbf{r}^*$.

The recovery is based on the optimal dual variable of the last correction step. If at least two entries in $\boldsymbol{\lambda}^*$ are equal, time-sharing may be required. In the case of equal entries in $\boldsymbol{\lambda}^*$, there exist multiple rate vectors $\mathbf{r} \in \mathcal{R}$ that are maximisers of a WsrMax problem with weight $\boldsymbol{\lambda}^*$ [4], and $\mathbf{r}^*$ is a convex combination of these points. In the case that all entries in $\boldsymbol{\lambda}^*$ are equal, all permutations π are optimal, resulting in $K!$ points $\mathbf{r}_{\mathrm{wsr}}(\boldsymbol{\lambda}^*, \pi)$. As a consequence, enumerating all $K!$ points first and then selecting the (at most) K points that are actually required to implement $\mathbf{r}^*$ are only feasible for small K. For larger K, an efficient method for identifying a set of relevant points is provided in [30].

If no two entries in $\boldsymbol{\lambda}^*$ are equal, the optimum encoding order π^* is uniquely defined, $\mathbf{r}^* = \mathbf{r}_{\mathrm{wsr}}(\boldsymbol{\lambda}^*, \pi^*)$, and $\mathbf{Q}^*$ maximises $(\boldsymbol{\lambda}^*)^{\mathrm{T}}\mathbf{r}(\mathbf{Q}, \pi^*)$, compare (13).

From an implementation viewpoint, entries in $\boldsymbol{\lambda}^*$ will usually not be exactly equal, even if the theoretical solution lies in a time-sharing region. As a result, time-sharing between users is declared if the difference between weights is below a certain threshold.

5.4. Coarse Projection. The proposed algorithm consists of two nested loops: a gradient-based outer loop and an inner loop for the correction step at each outer iteration. A significant reduction in computational complexity can be achieved if the required precision of the inner loop is adapted to the outer loop. In fact, the convergence of the outer loop is ensured by an increase in the cost function at each step, based on condition (20). The inner iteration generates rate vectors $\mathbf{r}^*(\boldsymbol{\lambda}^{(i)})$ during convergence to $\boldsymbol{\lambda}^*$. If $\mathbf{r}^*(\boldsymbol{\lambda}^{(i)})$ fulfills condition (20) and $\mathbf{r}^*(\boldsymbol{\lambda}^{(i)}) \in \widetilde{\mathcal{E}}$, the projection of $\widetilde{\mathbf{r}}$ on $\mathcal{C}$ is sufficiently good to yield an ascent step on $\widetilde{\mathcal{E}}$. In this case, the projection is aborted, and the outer loop continues with

$$\mathbf{r}^{(n+1)} = \mathbf{r}^*\left(\boldsymbol{\lambda}^{(i)}\right). \tag{43}$$

The resulting reduction in the number of inner iterations comes at the price of an evaluation of the function u at each inner iteration. As a result, the overall gain in terms of complexity clearly depends on the cost associated with evaluating u.

5.5. Boundary Points. So far, it has been assumed that at the optimal solution $\mathbf{r}^*$, all users have nonzero rate (i.e., $\mathbf{r}^* \in \widetilde{\mathcal{E}}$). If this assumption does not hold, the sequence $\{\mathbf{r}^{(n)}\}$ converges to a point on the boundary of $\mathcal{E}$, compare Section 5.6. Define

$$\mathcal{I}(\mathbf{r}) = \{k : r_k = 0\}. \tag{44}$$

The boundary of $\mathcal{E}$ is given by

$$\partial\mathcal{E} = \mathcal{E} \setminus \widetilde{\mathcal{E}} = \{\mathbf{r} \in \mathcal{E} : \mathcal{I}(\mathbf{r}) \neq \varnothing\}. \tag{45}$$

Observe that the boundary can be written as the union of K sets $\partial\mathcal{E}_{\{k\}}$, with

$$\partial\mathcal{E}_{\{k\}} = \{\mathbf{r} \in \mathcal{E} : \{k\} \subset \mathcal{I}(\mathbf{r})\}. \tag{46}$$

Finally, define a set $\mathcal{E}_{\{k\}}$ by removing the kth entry (which is zero) from all elements in $\partial\mathcal{E}_{\{k\}}$:

$$\mathcal{E}_{\{k\}} = \{\mathbf{x} \in \mathbb{R}^{K-1} : x_\ell = r_\ell,\ \ell \notin \{k\},\ \mathbf{r} \in \partial\mathcal{E}_{\{k\}}\}. \tag{47}$$

Note that the resulting set $\mathcal{E}_{\{k\}}$ is the efficient boundary of a capacity region of a $K-1$ user MIMO BC, with user k removed. It follows immediately that the interior $\mathcal{E}_{\{k\}}$ is again a differentiable manifold, now of dimension $K-1$. The boundary of $\mathcal{E}_{\{k\}}$ can be decomposed in the same manner, resulting in a set of $K-2$ dimensional manifolds, and so on. Accordingly, the set $\mathcal{E}_{\mathcal{D}}$, with $\mathcal{D} \subseteq \{1,\ldots,K\}$ corresponds to the efficient boundary of a capacity region of a $K-|\mathcal{D}|$ user MIMO BC, with users in $\mathcal{D}$ removed.

Accordingly, the general case is incorporated as follows. Denote by $\mathcal{A} = \{1,\ldots,K\} \setminus \mathcal{D}$ the set of active users. Only active users are considered in the optimisation, that is, replace K by $|\mathcal{A}|$ and let k be the index of the kth active user in all steps of the algorithm. If the sequence $\{\mathbf{r}^{(n)}\}$ converges to a point on the boundary of $\mathcal{E}_{\mathcal{D}}$, the users with zero entries in the rate vector are removed from $\mathcal{A}$ and assigned to $\mathcal{D}$. Initialise with $\mathcal{A} = \{1,\ldots,K\}$, $\mathcal{D} = \varnothing$, and $\mathbf{r}^{(0)} \in \widetilde{\mathcal{E}}$. With these modifications, the algorithm always operates on differentiable manifolds $\widetilde{\mathcal{E}}_{\mathcal{D}} \subset \mathbb{R}^{|\mathcal{A}|}$, with $\mathbf{r} \gg \mathbf{0}$ for all $\mathbf{r} \in \widetilde{\mathcal{E}}_{\mathcal{D}}$.

In practice, convergence to the boundary is detected as follows. If the rate $r_k^{(n)}$ of an active user falls below a threshold, and the projected utility gradient results in $r_k^{(n+1)} < r_k^{(n)}$, the user is deactivated. The decision to deactivate a user is based on the iterates and not on the limit point, thus the modified algorithm may lead to suboptimal results if a user is deactivated that actually has nonzero rate in the limit.

5.6. Convergence of the IEA Method. Concerning the convergence of the IEA method, two cases can be distinguished. In the first case, the sequence $\{\mathbf{r}^{(n)}\}$ converges to a point in $\widetilde{\mathcal{E}}$. In the second case, the sequence $\{\mathbf{r}^{(n)}\}$ converges to a point on the boundary of $\mathcal{E}$. According to Section 5.5, after removing the users with zero rate, the boundary itself is a $K-1$ dimensional manifold with boundary, and the algorithm converges in the interior or on the boundary of this manifold. The argument continues until the dimension of the manifold under consideration is 0. Thus, it suffices to consider the convergence behaviour in the interior of $\mathcal{E}_{\mathcal{D}}$, which, from the perspective of the algorithm, is equivalent to $\widetilde{\mathcal{E}}$—an open set equipped with a differentiable manifold structure.

Accordingly, the IEA method is globally convergent if convergence to a point $\mathbf{r}^* \in \widetilde{\mathcal{E}}$ implies that $\mathbf{r}^* \in \widetilde{\mathcal{E}}$ is a stationary point. Convergence can be proved using Zangwill's global convergence theorem [26]. Not all parameterisations, however, yield a convergent method. For the parameterisation defined in (28), global convergence (in the sense of the global convergence theorem) is proved in [31].

A more intuitive (and less rigorous) discussion of the convergence behaviour follows from considering the updates $\boldsymbol{\mu}^{(n+1)}$. Convergence to a point $\mathbf{r}^*$ implies

$$\boldsymbol{\mu}^{(n+1)} = t^{(n)} \nabla f_{\mathbf{r}^{(n)}}(\mathbf{0}) \longrightarrow 0. \tag{48}$$

Now assume that $\mathbf{r}^*$ is not a stationary point. This implies $\nabla f_{\mathbf{r}^{(n)}}(\mathbf{0}) \neq \mathbf{0}$, for all n, which, by (48), implies $t^{(n)} \to 0$. For the parameterisation defined in (28), such a sequence of step sizes results if the sequence of upper bounds $\bar{t}(\mathbf{r}^{(n)})$ converges to zero. This behaviour, however, only occurs if the sequence $\{\mathbf{r}^{(n)}\}$ converges to a point on the boundary of $\mathcal{E}$, which contradicts the assumption that $\mathbf{r}^* \in \widetilde{\mathcal{E}}$.

The theoretical convergence results based on Zangwill's global convergence theorem assume infinite precision. Theoretically, if $\nabla f_{\mathbf{r}^{(n)}}(\mathbf{0}) \neq \mathbf{0}$, it is always possible to find a step size $t > 0$ such that (20) holds. In a practical implementation of the IEA method, the parameterisation is evaluated numerically, in particular the correction step is a numerical solution of a convex optimisation problem. Due to the convexity of the correction problem, a high numerical precision can be achieved. Still, the inherent finite precision of the correction step sets a limit to the precision of the overall algorithm. This property underlines the importance of the coarse projection described in Section 5.4. The inner loop needs a tight convergence criterion in order to yield a high precision in cases where it is difficult to find an ascent step. In cases where an ascent step is easily found, however, it is not necessary to solve the problem to high precision. The latter case is detected by the coarse projection. Also note that the coarse projection does not impact the convergence behaviour in a negative way. The global convergence ensures that (theoretically) the algorithm does not get stuck at a nonstationary point. The coarse projection only comes into play if it is possible to move away from the current point.

It is clearly not guaranteed that a stationary point $\mathbf{r}^*$ maximises utility. Due to the fact that the proposed algorithm is an ascent method, however, $\mathbf{r}^*$ is a good solution in the sense that given an initial value $\mathbf{r}^{(0)}$, utility is either improved, or the algorithm converges at the first iteration and stays at $\mathbf{r}^{(0)}$, in this case requiring no extra computations. That is, any investment in terms of computational effort is rewarded with a gain in utility.

6. Monotonic Optimisation

The gradient-based approach presented in Section 5 converges to a stationary point of the optimisation problem, and cannot guarantee convergence to global optimality, as it relies on local information only.

The rate-space formulation (12) of the utility maximisation problem corresponds to the maximisation of a monotonic function (the utility function u) over a compact set in $\mathbb{R}_+^K$ (the capacity region $\mathcal{C}$), and hence is a monotonic optimisation problem [15], which can be solved to global optimality.

A basic problem of monotonic optimisation is the maximisation of a monotonic function over a compact normal set [15]. A subset $\mathcal{S}$ of $\mathbb{R}_+^K$ is said to be *normal* in $\mathbb{R}_+^K$ (or briefly, normal), if $\mathbf{x} \in \mathcal{S}$, $\mathbf{0} \leq \mathbf{y} \leq \mathbf{x} \Rightarrow \mathbf{y} \in \mathcal{S}$. The capacity region $\mathcal{C}$ is normal: any rate vector $\mathbf{r}'$ that is smaller than an achievable rate vector $\mathbf{r}$ is also achievable. Thus, $\mathcal{C}$ is a compact normal set and the rate-space problem (12) is a basic problem of monotonic optimisation.

6.1. Polyblock Algorithm. The basic algorithm for solving monotonic optimisation problems is the so-called *polyblock*

algorithm. A polyblock is simply the union of a finite number of hyperrectangles in $\mathbb{R}_+^K$. Given a discrete set $\mathcal{V} \subset \mathbb{R}_+^K$, a polyblock $\mathcal{P}(\mathcal{V})$ is defined as

$$\mathcal{P}(\mathcal{V}) = \bigcup_{\mathbf{v}\in\mathcal{V}} \{\mathbf{r} \in \mathbb{R}_+^K,\ \mathbf{r} \leq \mathbf{v}\}. \tag{49}$$

The set $\mathcal{V}$ contains the vertices of the polyblock $\mathcal{P}(\mathcal{V})$.

Due to the fact that $\mathcal{C}$ is a compact normal subset of $\mathbb{R}_+^K$, there exists a set $\mathcal{V}^{(0)}$ such that $\mathcal{C} \subseteq \mathcal{P}(\mathcal{V}^{(0)})$. Moreover, starting with $n = 0$, either $\mathcal{C} = \mathcal{P}(\mathcal{V}^{(n)})$ or there exists a discrete set $\mathcal{V}^{(n+1)} \subset \mathbb{R}_+^K$ such that

$$\mathcal{C} \subseteq \mathcal{P}(\mathcal{V}^{(n+1)}) \subset \mathcal{P}(\mathcal{V}^{(n)}). \tag{50}$$

In other words, the polyblocks $\mathcal{P}(\mathcal{V}^{(n)})$ represent an iteratively refined outer approximation of the capacity region.

Consider the problem of maximising utility over the polyblock $\mathcal{P}(\mathcal{V}^{(n)})$:

$$\max_{\mathbf{r}\in\mathcal{P}(\mathcal{V}^{(n)})} u(\mathbf{r}). \tag{51}$$

Let $\check{\mathbf{r}}^{(n)}$ denote a maximiser of problem (51), Due to the monotonicity of u, $\check{\mathbf{r}}^{(n)} \in \mathcal{V}^{(n)}$, that is, the maximum of a monotonic function over a polyblock is attained on one of the vertices [15]. Due to the fact that the vertex set of a polyblock is discrete, problem (51) can be solved to global optimality by searching over all $\mathbf{v} \in \mathcal{V}^{(n)}$.

If $\check{\mathbf{r}}^{(n)} \in \mathcal{E}$, the globally optimal rate vector is found. In general, however, $\check{\mathbf{r}}^{(n)}$ will lie outside the capacity region, due to the fact that the polyblock represents an outer approximation. Denote by $\mathbf{y}^{(n)} \in \mathcal{E}$ the intersection between $\mathcal{E}$ and the line segment connecting the origin with $\check{\mathbf{r}}^{(n)}$. Let $\widehat{\mathbf{r}}^{(n)}$ denote the best intersection point computed so far, that is,

$$\widehat{\mathbf{r}}^{(n)} = \mathbf{y}^{(\ell^*)}, \quad \ell^* = \arg\max_{\ell\in\{1,\ldots,n\}} u(\mathbf{y}^{(\ell)}). \tag{52}$$

Moreover, let u^* denote the global maximum of (12). It follows that

$$u(\widehat{\mathbf{r}}^{(n)}) \leq u^* \leq u(\check{\mathbf{r}}^{(n)}). \tag{53}$$

Intuitively, as the outer approximation of $\mathcal{C}$ by a polyblock is refined at each step, $u(\check{\mathbf{r}}^{(n)})$ eventually converges to u^*. Due to the continuity of u, this convergence also holds for $\widehat{\mathbf{r}}^{(n)}$, that is, $\widehat{\mathbf{r}}^{(n)}$ converges to a global maximiser of u. See [15] for a rigorous proof. According to (53), an ϵ-optimal solution is found if $u(\widehat{\mathbf{r}}^{(n)}) \geq u(\check{\mathbf{r}}^{(n)}) - \epsilon$.

One possible method to construct a sequence of polyblocks $\mathcal{P}(\mathcal{V}^{(n)})$ that satisfies (50) is as follows [15]. Define

$$\mathcal{K}(\mathbf{r}) = \{\mathbf{x} \in \mathbb{R}_+^K : x_k > r_k,\ k \notin \mathcal{U}(\mathbf{r})\}, \tag{54}$$

with $\mathcal{U}(\mathbf{r})$ as defined in (44). Clearly, $\widehat{\mathbf{r}}^{(n)} \in \mathcal{E}$ implies $\mathcal{K}(\widehat{\mathbf{r}}^{(n)}) \cap \mathcal{C} = \varnothing$. Thus, $\mathcal{K}(\widehat{\mathbf{r}}^{(n)})$ can be removed from $\mathcal{P}(\mathcal{V}^{(n)})$ without removing any achievable rate vector. Moreover, if $\check{\mathbf{r}}^{(n)} \notin \mathcal{E}$,

$$\mathcal{K}(\widehat{\mathbf{r}}^{(n)}) \cap \mathcal{P}(\mathcal{V}^{(n)}) \supset \{\check{\mathbf{r}}^{(n)}\} \supset \varnothing, \tag{55}$$

thus by removing $\mathcal{K}(\widehat{\mathbf{r}}^{(n)})$, a tighter approximation results. Finally, $\mathcal{P}(\mathcal{V}^{(n)}) \setminus \mathcal{K}(\widehat{\mathbf{r}}^{(n)})$ is again a polyblock [15]. To summarise, the desired rule for constructing a sequence of polyblocks that satisfies (50) is

$$\mathcal{P}(\mathcal{V}^{(n+1)}) = \mathcal{P}(\mathcal{V}^{(n)}) \setminus \mathcal{K}(\widehat{\mathbf{r}}^{(n)}). \tag{56}$$

The rules for computing the corresponding vertex set $\mathcal{V}^{(n+1)}$ are provided in [15].

6.2. Intersection with $\mathcal{E}$. If the polyblock algorithm is applied to the rate-space problem (12), the only step in the algorithm in which the intricate properties of the capacity region $\mathcal{C}$ come into play is the computation of the intersection between $\mathcal{E}$ and the line connecting the origin with $\check{\mathbf{r}}^{(n)}$. Comparing the correction step of the IEA algorithm from Section 5.2 with the computation of the intersection point, it turns out that both operations are almost identical, only the line whose intersection with $\mathcal{E}$ is computed is different. As a result, the Lagrangian-based algorithm from Section 5.2 can also be used to compute the intersection point, by setting

$$\widetilde{\mathbf{r}} = \check{\mathbf{r}}^{(n)}, \qquad \mathbf{n} = \check{\mathbf{r}}^{(n)}. \tag{57}$$

In Section 5, it was stated that the most complex step in each iteration of the IEA method is the correction step. Similar results hold for the polyblock algorithm. At each iteration, the main complexity lies in the computation of the intersection point. Due to the similarity between IEA's correction step and the computation of the intersection point in the polyblock algorithm, the complexity of both approaches can be compared by comparing the number of gradient iterations with the number of polyblocks generated until a sufficiently tight outer approximation is found. The convergence properties of the polyblock algorithm are only asymptotic [15]—thus, a relatively high complexity of the polyblock algorithm can be expected. This expectation is confirmed by simulation results; see Section 8.

6.3. Implementation Issues. The presentation of the polyblock algorithm in Section 6.1 closely follows [15]. In this basic version, simulations showed very slow convergence of the algorithm, due to the fact that close to regions on the boundary where at least on rate gets close to zero, a large number of iterations are needed until a significant refinement results. A similar behaviour is reported in [32]. Following [32], the convergence speed of the algorithm can be significantly improved by modifying the direction of the line whose intersection with $\mathcal{E}$ defines the next iterate $\mathbf{y}^{(n)}$. Computationally, this is achieved by setting $\mathbf{n} = \check{\mathbf{r}}^{(n)} + \mathbf{a}$, $\mathbf{a} \in \mathbb{R}_+^K$ in the algorithm from Section 5.2.

An initial vertex set $\mathcal{V}^{(0)}$ can be determined as follows. Define a rate vector $\mathbf{v} \in \mathbb{R}_+^K$ whose kth entry v_k corresponds to the maximum rate achievable for user k. Then, $\mathcal{V}^{(0)} = \{\omega\mathbf{v}\}$ with $\omega \geq 1$ defines a polyblock that contains the capacity region.

7. Dual Decomposition

For concave utilities, a dual approach to solve the utility maximisation problem in the MIMO BC was recently proposed in [14]. The algorithm in [14] represents an application of the dual decomposition [10]. Similar to the gradient-based method developed in Section 5, the solution is found in two steps. First, an optimal rate vector $\mathbf{r}^*$ is found by operating in the rate space; second, the optimal parameters are derived from $\mathbf{r}^*$.

In the first step, problem (12) is modified by introducing additional variables:

$$\max_{\mathbf{r},\mathbf{s}} u(\mathbf{s}) \quad \text{s.t. } \mathbf{0} \le \mathbf{s} \le \mathbf{r},\ \mathbf{r} \in \mathcal{C}. \tag{58}$$

The dual function is chosen as

$$g(\boldsymbol{\lambda}) = \underbrace{\max_{\mathbf{s}\ge\mathbf{0}} u(\mathbf{s}) - \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{s}}_{g_{\mathrm{A}}(\boldsymbol{\lambda})} + \underbrace{\max_{\mathbf{r}\in\mathcal{C}} \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{r}}_{g_{\mathrm{P}}(\boldsymbol{\lambda})}. \tag{59}$$

Evaluating the dual function at $\boldsymbol{\lambda}$ results in two decoupled subproblems, computing $g_{\mathrm{A}}(\boldsymbol{\lambda})$ and $g_{\mathrm{P}}(\boldsymbol{\lambda})$ by maximising over the primal variables $\mathbf{s}$ and $\mathbf{r}$, respectively. Computing $g_{\mathrm{P}}(\boldsymbol{\lambda})$ is again a WsrMax problem.

The optimal dual variable is found by minimising the dual function with respect to $\boldsymbol{\lambda}$. The dual function is always convex, regardless of the properties of the utility function u [16].

If the utility function u is concave, strong duality holds, and the optimal primal solution $\mathbf{r}^*$ can be recovered from the dual solution by employing standard methods for primal recovery, as in [14]. Also, for concave u, efficient methods exist to find a set of corner points that implement $\mathbf{r}^*$ in the case of time-sharing optimality [30].

Being entirely based on Lagrange duality, a nonconcave utility poses significant problems to the dual decomposition. Most importantly, recovering an *optimal* primal solution $(\mathbf{r}^*, \mathbf{s}^*)$ from the dual solution is, in general, no longer possible. Moreover, the schemes for recovering all parameters $\mathbf{x}_{\mathrm{P}}$ of a time-sharing solution rely on strong duality to hold [30]. For nonconcave u, however, strong duality cannot be assumed to hold. In fact, simulation results in Section 8 show a significant duality gap in the scenario under consideration.

As a result, for nonconcave u, the following *heuristic* is used to obtain a primal feasible solution $(\hat{\mathbf{r}}, \hat{\mathbf{s}})$. Given the optimal dual variable $\boldsymbol{\lambda}^*$, choose $\hat{\mathbf{r}} = \mathbf{r}_{\mathrm{wsr}}(\boldsymbol{\lambda}^*, \pi^*)$, where π^* is any optimal encoding order. Moreover, let $\hat{\mathbf{s}} = \hat{\mathbf{r}}$. An upper bound on the loss incurred by this approximation follows immediately from weak duality. Let u^* denote the (unknown) maximum utility value. By weak duality, $g(\boldsymbol{\lambda}^*) \ge u^*$, thus $u^* - u(\hat{\mathbf{r}}) \le g(\boldsymbol{\lambda}^*) - u(\hat{\mathbf{r}})$. The tightness of this bound clearly depends on the duality gap, which is not known.

8. Simulation Results

Utility maximisation in a $K = 3$ user Gaussian MIMO broadcast channel with $N = 6$ transmit antennas and $M_k = 2$ receive antennas per user is simulated. The channels $\mathbf{H}_k$ are i.i.d. unit-variance complex Gaussian. Furthermore, the maximum transmit power is $P_{\mathrm{tr}} = 10$. To obtain rates in Kbps, rates are multiplied by a bandwidth factor $W = 60$ kHz.

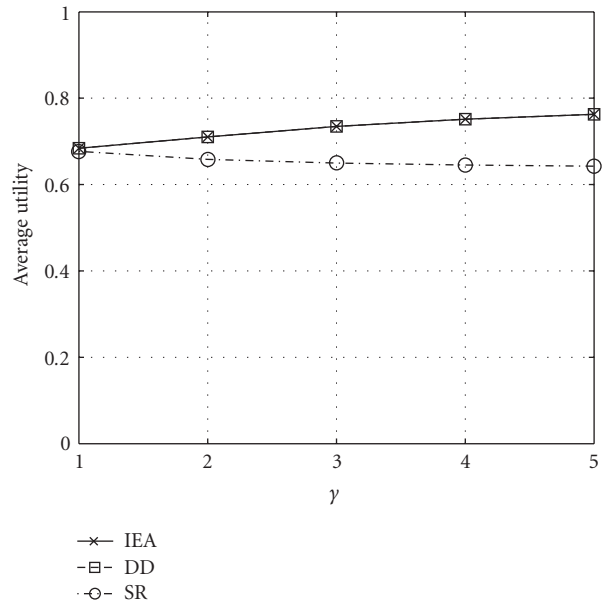

Figure 2: Average utility (concave utilities).

In the simulations, the utility u is given by a weighted sum of the users' utilities u_k:

$$u(\mathbf{r}) = \sum_{k=1}^{K} w_k u_k(r_k). \tag{60}$$

The IEA method always uses a sum-rate maximising rate vector as initial point $\mathbf{r}^{(0)}$. The results are averaged over 1000 channel realisations.

Two different models for the users' utilities u_k are considered: a concave logarithmic utility and a nonconcave sigmoidal utility.

8.1. Concave Utility. The logarithmic utility function is defined as

$$u_k(r_k) = b \ln\left(1 + c^{-1} r_k\right), \tag{61}$$

with constants b, c. In the simulations, $c = 40$ Kbps and b is chosen such that $u_k(1000\,\text{Kbps}) = 1$. The weights w_k are chosen according to the following scheme:

$$\begin{aligned} \boldsymbol{\omega} &= \left(1 \quad \gamma \quad \gamma^2\right), \\ \mathbf{w} &= \frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|_1}, \end{aligned} \tag{62}$$

with $\gamma \in \{1, \ldots, 5\}$. Figure 2 shows the average utility for the case of logarithmic utility functions. What is shown is the average utility for the gradient-based approach (IEA), for the dual decomposition (DD), and, as a reference, the average utility obtained by using for transmission the sum-rate (SR) maximising rate vector that corresponds to encoding order $\pi = [1 \ \ 2 \ \ 3]$.

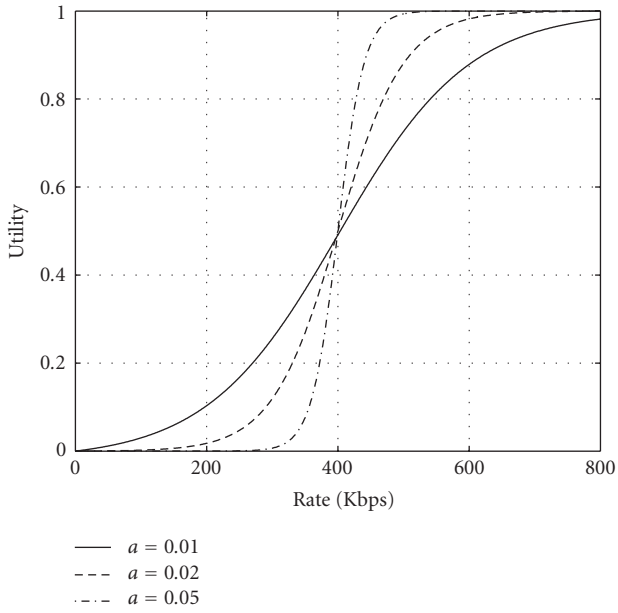

Figure 3: Sigmoid utility function, $b = 400$ Kbps.

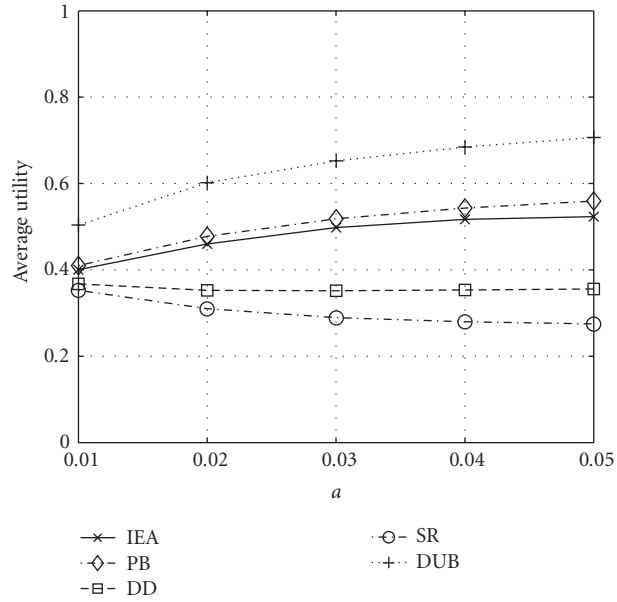

Figure 4: Average utility (sigmoidal utilities).

Due to the fact that the utility maximisation problem is convex, both IEA and DD achieve identical performance. Moreover, for identical weights w_k, cross-layer optimisation does not provide a significant gain compared to the sum-rate maximising strategy. The larger the difference between the users' weights, the larger the gain achieved by cross-layer optimisation. This result is not surprising, as for asymmetric setups, it is more important to adapt the physical layer to the characteristics of the upper layers. Moreover, the decay of the logarithmic utility function is rather moderate around the optimal rate vector, therefore a maximiser of the weighted sum-rate is almost optimal for equal weights.

8.2. Nonconcave Utility. The nonconcave utility model is adopted from [8]. For each user k, the following sigmoidal utility function is used:

$$u_k(r_k) = c_k\left(\frac{1}{1+\exp\left(-a_k(r_k-b_k)\right)}+d_k\right), \tag{63}$$

where c_k and d_k are used to normalise u_k such that $u_k(0) = 0$ and $u_k(\infty) = 1$. The steepness of the transition between the minimum value and the maximum value is controlled by the parameter a_k, whereas b_k determines the inflection point of the utility curve (cf. Figure 3). In the simulations, $a_k = a\,\text{Kbps}^{-1}$, and a is varied in a range between 0.01 and 0.05, modelling different degrees of elasticity of the applications. For each channel realisation, the constant b_k of each user is chosen randomly in the interval [300 Kbps, 500 Kbps] according to a uniform distribution. Choosing the b_k randomly can be understood as a model for fluctuations in the data rate requirements of the users over time, that is, transmission of a video source with varying scene activity. All users have equal weight $w_k = 1/K$.

Figure 4 shows the average utility for the case of sigmoidal utility functions. What is shown is the average utility for the gradient-based approach (IEA), the polyblock algorithm (PB), the dual decomposition (DD), and the sum-rate (SR) maximising rate vector. In addition, the average minimum value of the dual function in the dual decomposition approach is shown (DUB). The PB algorithm finds the global maximum for each realisation. As a result, the PB curve gives the maximum achievable average utility. In terms of average utility, the performance of the IEA method is close to optimal. It can be concluded that for the system setup under consideration, the IEA method succeeds in finding a stationary point which is identical or close to the global maximum for most realisations. In contrast, the dual decomposition-based method does not find a good rate vector in most cases. The poor performance of the computationally simple SR strategy emphasises the need for cross-layer optimisation. In particular, the performance gain achieved by both PB and IEA increases with a. This behaviour can be explained as follows. With increasing a, the interval in which the utility function makes a transition from small to large values becomes smaller. Therefore, it becomes more and more important to adapt the physical layer parameters to the utility characteristics.

The results in Figure 4 also show that the dual upper bound (DUB) obtained from the dual decomposition is rather loose. This implies that there is a significant duality gap in most cases.

8.3. Complexity Analysis. If average utility is the only figure of merit, the polyblock algorithm is obviously superior to all other approaches. From a practical viewpoint, a second metric of interest is the computational complexity of the different approaches. In the following, the utility-complexity tradeoffs provided by the different approaches are investigated. All results are for the case of sigmoidal utility functions.

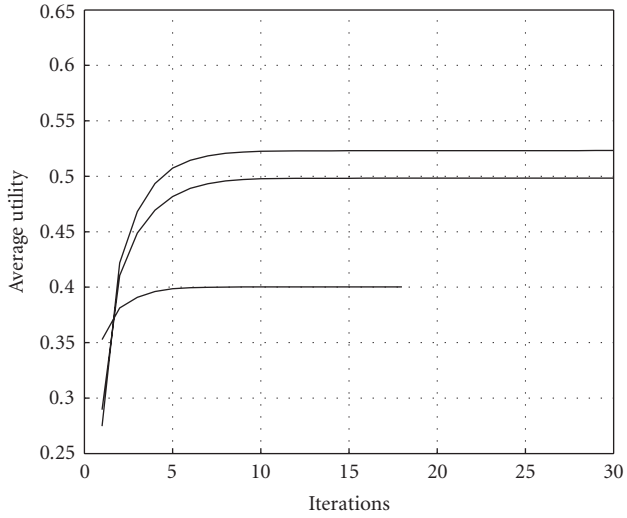

Figure 5: Average utility versus number of iterations, IEA method.

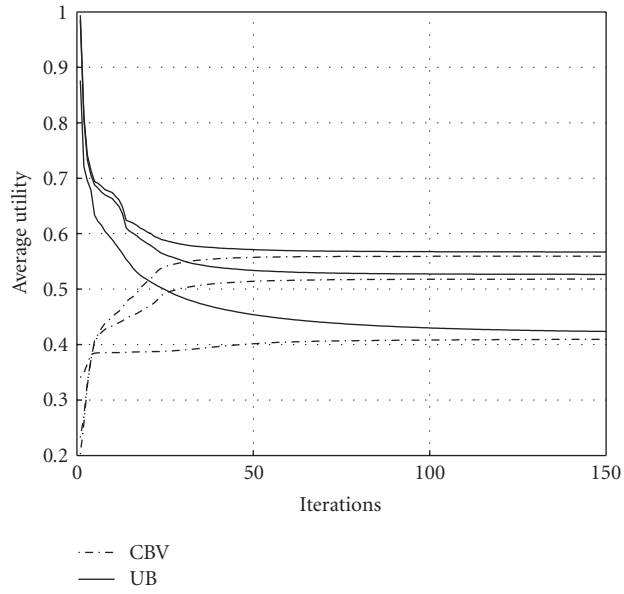

Figure 6: Average utility versus number of iterations, PB algorithm.

In Figure 5, average utility is plotted versus the number of iteration for the IEA method. The plot shows three graphs, corresponding to three different values of the steepness parameter a: $a \in \{0.01, 0.03, 0.05\}$. Note that the rightmost points of each graph correspond to the average utility value in Figure 4. Only the gradient-based outer iterations defined in (21)–(23) are counted, the inner iterations in the correction step are neglected. Figure 5 shows that the IEA method needs between five and 10 iterations to get close to the maximum achieved utility value.

In Figure 6, average utility is plotted versus the number of iteration for the polyblock algorithm. The plot shows three pairs of graphs, with each pair corresponding to a different value of the steepness parameter a: $a \in \{0.01, 0.03, 0.05\}$. Each pair consists of two graphs, one showing the average of the current best utility value $u(\hat{\mathbf{r}}^{(n)})$ (CBV, dash-dotted line), the other showing the average of the upper bound $u(\check{\mathbf{r}}^{(n)})$ (UB, solid line). Depending on the parameter a, between 50 to 75 iterations are needed until the current best value is close to the global maximum. Recall from Section 6.1 that the convergence criterion for the PB algorithm is based on the difference between $u(\hat{\mathbf{r}}^{(n)})$ and $u(\check{\mathbf{r}}^{(n)})$. Figure 6 shows that a large number of iterations may be required until convergence is declared, due to the relatively slow convergence of the upper bound.

In both Figures 5 and 6, the number of inner iterations required in the correction step and the computation of the intersection point, respectively, are not counted. In each inner iteration, a WsrMax problem is solved. Moreover, a WsrMax problem is also solved at each iteration of the dual decomposition. Accordingly, all three approaches can be compared based on the number of calls to the WsrMax subroutine. Figure 7 shows the average utility that is achieved if the maximum number of calls to WsrMax is limited to a value maxcall, with maxcall increased in steps of 10 calls. Again, three groups of graphs are shown, each group corresponding to a value of a, with $a \in \{0.01, 0.03, 0.05\}$. As an example, the results show that the dual decomposition needs between 10 and 20 iterations until convergence (to a clearly suboptimal solution). Of particular interest are the cross-over points between IEA method and PB algorithm. For $a = 0.05$, the cross-over point is at maxcall $= 300$, that is, only if more than a maximum of 300 calls to WsrMax are feasible does the PB algorithm outperform the IEA method. Moreover, for small values of maxcall, the IEA method provides significantly larger average utility.

9. Conclusions

Two methods for solving the nonconcave utility maximisation problem in the MIMO broadcast channel are proposed: a gradient-based method that converges to a locally optimal solution, and an approach based on monotonic optimisation that yields the global optimum. Due to the structure of the MIMO BC capacity region, a direct optimisation in terms of the physical layer parameters transmit covariance matrices and encoding order is not feasible. Thus, as an intermediate step, both methods first determine an optimal rate vector. The optimal physical layer parameter setup, which may include a time-sharing solution, is then obtained from this rate vector. For both methods, the formulation of the utility maximisation problem in the rate space represents a key step. The IEA method exploits the differentiable manifold structure of the efficient boundary of the capacity region, while the polyblock algorithm relies on the fact that maximising utility over the set of achievable rate vectors represents a monotonic optimisation problem.

The polyblock algorithm provides globally optimal performance, at the price of a relatively high computational complexity. From a practical viewpoint, the proposed IEA method provides an attractive tradeoff between utility performance and computational complexity. In the simulation setup used in this work, the average utility achieved by

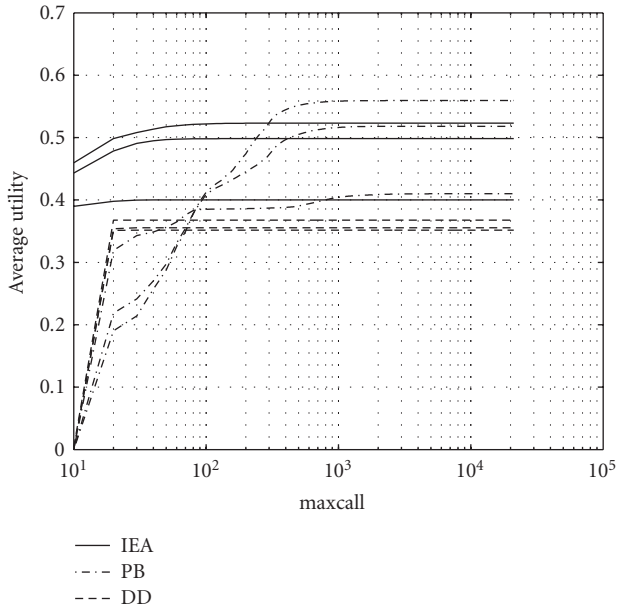

Figure 7: Average utility versus maximum number of WsrMax calls.

the IEA method is close to optimal, at significantly lower complexity than the polyblock algorithm.

Throughout this work, it is assumed that users' rates are the only relevant performance metrics of the physical layer, implying that rate cannot be traded for delay and reliability. In a more general setup, more than one performance metric per user may be required to characterise the physical layer, corresponding to a utility function that is a function of all these metrics [7]. Concerning the results presented in this work, this would clearly impact the mapping from physical layer parameters to set of achievable performance vectors. The methods proposed in this work, however, would still be applicable, provided the structural assumptions of each method are still met (i.e., the utility function is monotone in all physical layer metrics, the set of achievable performance vectors is compact and, in case of the IEA method, can be equipped with a differentiable manifold structure). While the capacity region is convex, it is not clear whether a generalised achievable region can still be assumed to be convex. This observation represents a further motivation for an optimisation framework that does not rely on the assumption of convexity.

References

[1] G. Caire and S. Shamai, "On the achievable throughput of a multiantenna Gaussian broadcast channel," *IEEE Transactions on Information Theory*, vol. 49, no. 7, pp. 1691–1706, 2003.

[2] P. Viswanath and D. N. C. Tse, "Sum capacity of the vector Gaussian broadcast channel and uplink-downlink duality," *IEEE Transactions on Information Theory*, vol. 49, no. 8, pp. 1912–1921, 2003.

[3] S. Vishwanath, N. Jindal, and A. Goldsmith, "Duality, achievable rates, and sum-rate capacity of Gaussian MIMO broadcast channels," *IEEE Transactions on Information Theory*, vol. 49, no. 10, pp. 2658–2668, 2003.

[4] H. Viswanathan, S. Venkatesan, and H. Huang, "Downlink capacity evaluation of cellular networks with known-interference cancellation," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 5, pp. 802–811, 2003.

[5] N. Jindal and A. Goldsmith, "Dirty-paper coding versus TDMA for MIMO broadcast channels," *IEEE Transactions on Information Theory*, vol. 51, no. 5, pp. 1783–1794, 2005.

[6] H. Weingarten, Y. Steinberg, and S. Shamai, "The capacity region of the Gaussian multiple-input multiple-output broadcast channel," *IEEE Transactions on Information Theory*, vol. 52, no. 9, pp. 3936–3964, 2006.

[7] S. Shenker, "Fundamental design issues for the future Internet," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 7, pp. 1176–1188, 1995.

[8] J.-W. Lee, R. R. Mazumdar, and N. B. Shroff, "Non-convex optimization and rate control for multi-class services in the Internet," *IEEE/ACM Transactions on Networking*, vol. 13, no. 4, pp. 827–840, 2005.

[9] M. Fazel and M. Chiang, "Network utility maximization with nonconcave utilities using sum-of-squares method," in *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference (CDC-ECC '05)*, pp. 1867–1874, Seville, Spain, December 2005.

[10] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: a mathematical theory of network architectures," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255–312, 2007.

[11] F. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, no. 1, pp. 33–37, 1997.

[12] X. Lin, N. B. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1452–1463, 2006.

[13] S. Stanczak, M. Wiczanowski, and H. Boche, *Resource Allocation in Wireless Networks: Theory and Algorithms*, Lecture Notes in Computer Science 4000, Springer, Berlin, Germany, 2006.

[14] J. Liu, Y. T. Hou, and H. D. and Sherali, "Routing and power allocation for MIMO-based ad hoc networks with dirty paper coding," *IEEE International Conference on Communications (ICC '08)*, pp. 2859–2864, May 2008.

[15] H. Tuy, "Monotonic optimization: problems and solution approaches," *SIAM Journal on Optimization*, vol. 11, no. 2, pp. 464–494, 2000.

[16] D. Bertsekas, A. Nedic, and A. Ozdaglar, *Convex Analysis and Optimization*, Athena Scientific, Belmont, Mass, USA, 2003.

[17] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable H.264/MPEG4-AVC extension," in *Proceedings of the IEEE International Conference on Image Processing (ICIP '06)*, pp. 161–164, Atlanta, Ga, USA, October 2006.

[18] D. N. C. Tse and S. V. Hanly, "Multiaccess fading channels—part I: polymatroid structure, optimal resource allocation and throughput capacities," *IEEE Transactions on Information Theory*, vol. 44, no. 7, pp. 2796–2815, 1998.

[19] M. Kobayashi and G. Caire, "An iterative water-filling algorithm for maximum weighted sum-rate of Gaussian MIMO-BC," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1640–1646, 2006.

[20] R. Böhnke and K.-D. Kammeyer, "Weighted sum rate maximization for the MIMO-downlink using a projected conjugate

gradient algorithm," in *Proceedings of the International Workshop on Cross Layer Design (IWCLD '07)*, pp. 82–85, Jinan, China, September 2007.

[21] D. G. Luenberger, "The gradient projection method along geodesics," *Management Science*, vol. 18, no. 11, pp. 620–631, 1972.

[22] D. Gabay, "Minimizing a differentiable function over a differential manifold," *Journal of Optimization Theory and Applications*, vol. 37, no. 2, pp. 177–219, 1982.

[23] J. H. Manton, "Optimization algorithms exploiting unitary constraints," *IEEE Transactions on Signal Processing*, vol. 50, no. 3, pp. 635–650, 2002.

[24] J. B. Rosen, "The gradient projection method for nonlinear programming—part II: nonlinear constraints," *SIAM Journal on Applied Mathematics*, vol. 9, no. 4, pp. 514–553, 1961.

[25] W. Boothby, *An Introduction to Differentiable Manifolds and Riemannian Geometry*, Academic Press, New York, NY, USA, 1975.

[26] W. Zangwill, *Nonlinear Programming: A Unified Approach*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1969.

[27] J. H. Manton, "On the various generalisations of optimisation algorithms to manifolds," in *Proceedings of the 6th International Symposium on Mathematical Theory of Networks and Systems (MTNS '04)*, Leuven, Belgium, July 2004.

[28] D. G. Luenberger, *Linear and Nonlinear Programming*, Addison-Wesley, Reading, Mass, USA, 2nd edition, 1984.

[29] J.-L. Goffin and J.-P. Vial, "Convex nondifferentiable optimization: a survey focussed on the analytic center cutting plane method," Tech. Rep. 99.02, Logilab, Geneva, Switzerland, 1999.

[30] J. Brehmer, Q. Bai, and W. Utschick, "Time-sharing solutions in MIMO broadcast channel utility maximization," in *Proceedings of the International ITG Workshop on Smart Antennas (WSA '08)*, pp. 153–156, Vienna, Austria, February 2008.

[31] M. Riemensberger and W. Utschick, "Gradient descent on differentiable manifolds," Tech. Rep. MSV-2008-1, Technische Universität München, Munich, Germany, 2008.

[32] H. Tuy, F. Al-Khayyal, and P. Thach, "Monotonic optimization: branch and cut methods," in *Essays and Surveys in Global Optimization*, C. Audet, P. Hansen, and G. Savard, Eds., pp. 39–78, Springer, New York, NY, USA, 2005.