A SYMBOL-WISE ORDER OPTIMIZATION FOR SUCCESSIVE PRECODING

Michael Joham, David A. Schmidt, Hans Brunner, and Wolfgang Utschick

Associate Institute for Signal Processing, Munich University of Technology
Theresienstr. 90, 80333 Munich, Germany
E-Mail: {joham,dschmidt,utschick}@tum.de

ABSTRACT

We propose a new approach to computing the perturbation signal for the vector broadcast channel with modulo receivers. Since the optimum *vector precoding* (VP) has prohibitive complexity, we focus on successive precoding, also known as *Tomlinson Harashima precoding* (THP). The performance of THP strongly depends on the precoding order which is usually optimized for a block of symbols. We show that a large part of the performance loss of THP compared to VP is caused by the precoding order and not by the successive computation of the perturbation signal. Based on the observation that especially the index of the data stream precoded last affects the performance of THP, we develop a new variant of THP with a complexity polynomial in the number of receivers that computes the precoding order per symbol and leads to results very close to those of VP which has a complexity exponential in the number of receivers.

1. INTRODUCTION

When the decentralized receivers in the vector broadcast setup are equipped with modulo operators, the transmitter has the freedom to add a perturbation signal to the data signal prior to the linear transformation which gives the transmit signal. These degrees of freedom due to the modulo operators are optimally utilized by VP [1, 2] which finds the perturbation signal by a complete closest point search in a lattice. The computation of the linear transformation necessary for VP only has to be performed once per block of symbols and has the same complexity as that for linear precoding (e.g., [3, 4]). Since the closest point search in a lattice has a complexity exponential in the dimensionality of the lattice (e.g., [5]), VP has a per-symbol complexity that is exponential in the number of receivers and is thus prohibitive.

In the case of THP, the elements of the perturbation signal are computed successively by means of a feedback loop with a modulo operator [6, 7, 8, 9, 10]. Due to the successive computation, THP strongly depends on the precoding order (e.g., [7, 8, 9]) which is computed once per block of symbols based on an assumption on the statistics of the output signal of the transmitter's modulo operator (e.g., [10]). The perturbation signal of THP is inferior to that found by VP because of the successive nature of THP. However, THP has a substantially lower complexity per symbol than VP, as the application of the THP filters has a complexity quadratic (polynomial) in the number of receivers. In [8], an algorithm for the computation of the THP filters was presented which has the same order of complexity as that of linear precoding.

Our approach to THP is different. We do not compute the precoding order once per block of symbols. Instead, we propose to compute the precoding order for every data symbol separately, since the transmitter knows the data signal. Motivated by the result that especially the index of the data stream precoded last has a strong influence on the performance of THP (e.g., [8]), we propose the following procedure per data vector. For all possible values of the index of the data stream which is precoded last, we compute the MSEs of THP for the given data vector. Then, the transmit signal for that particular data vector is computed based on the precoding order leading to the minimum MSE.

After introducing the system model in Section 2, we review VP and standard THP based on the *minimum mean square error* (MMSE) criterion in Sections 3 and 4, respectively. The proposed THP with a precoding order optimized symbol-wise is presented in Section 5 and the complexity of the investigated precoding schemes is discussed in Section 6.1. The simulation results in Section 6 show that the performance of our symbol-wise optimized THP is close to that of VP.

Notation: Vectors and matrices are denoted by lower case bold and capital bold letters, respectively. We use $\mathrm{E}[\bullet]$, $\mathrm{Re}(\bullet)$, $\mathrm{Im}(\bullet)$, $\mathrm{tr}(\bullet)$, $(\bullet)^{\mathrm{T}}$, $(\bullet)^{\mathrm{H}}$, and $\|\bullet\|_2$ for expectation, real part of the argument, imaginary part, trace of a matrix, transposition, conjugate transposition, and Euclidian norm, respectively. The floor operator, which gives the largest integer smaller than or equal to the argument, is denoted by $\lfloor\bullet\rfloor$. All random sequences are assumed to be zero-mean and stationary. The covariance matrix of the vector random process $\boldsymbol{x}[n]$ is denoted by $\boldsymbol{R}_{\boldsymbol{x}} = \mathrm{E}[\boldsymbol{x}[n]\boldsymbol{x}^{\mathrm{H}}[n]]$, whereas the variance of the scalar random process $y[n]$ is $\sigma_y^2 = \mathrm{E}[|y[n]|^2]$. The $N \times N$ identity matrix is $\mathbf{I}_N$, its i-th column is $\boldsymbol{e}_i$. We use $\mathbf{0}_{N\times M}$ and $\mathbf{0}_N$ for the $N \times M$ zero matrix and the N-dimensional zero vector, respectively.

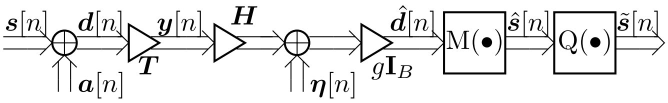

Fig. 1. System Model with Modulo Receivers.

2. SYSTEM MODEL

A perturbation signal $\boldsymbol{a}[n] \in \mathbb{M}^B$ is added to the data signal $\boldsymbol{s}[n] = [s_1[n], \ldots, s_B[n]]^{\mathrm{T}}$, where $s_b[n] \in \mathbb{A}$ is the data stream for receiver b. The sum $\boldsymbol{d}[n]$ is linearly transformed by $\boldsymbol{T} \in \mathbb{C}^{N \times B}$ to get the transmit signal $\boldsymbol{y}[n] \in \mathbb{C}^N$ (see Fig. 1). The modulation alphabet is $\mathbb{A}$ and the lattice [11] corresponding to the receivers' modulo operators $\mathrm{Mod}(x) = x - \tau\lfloor \mathrm{Re}(x)/\tau + 1/2\rfloor - \mathrm{j}\,\tau\lfloor \mathrm{Im}(x)/\tau + 1/2\rfloor$ is denoted by $\mathbb{M} = \tau\mathbb{Z} + \mathrm{j}\,\tau\mathbb{Z}$. The fundamental Voronoi region of $\mathbb{M}$ is $\mathbb{V}$, i.e., $\mathrm{Mod}(\bullet) \in \mathbb{V}$. To avoid ambiguities, we require that the modulo constant τ is sufficiently large to ensure that $\mathbb{A} \subset \mathbb{V}$ (e.g., [12]). For notational brevity, the modulo operators of all receivers are comprised in $\mathrm{M}(\bullet) \in \mathbb{V}^B$ which is defined element-wise, i.e., $\mathrm{M}(\boldsymbol{x}) = [\mathrm{Mod}(x_1), \ldots, \mathrm{Mod}(x_B)]^{\mathrm{T}}$, where x_k is the k-th element of the vector $\boldsymbol{x} \in \mathbb{C}^B$.

The transmit signal $\boldsymbol{y}[n]$ propagates over the channel $\boldsymbol{H}^{B \times N}$, of which the b-th row is the vector channel from the N channel inputs to receiver b, and is perturbed by the additive B-dimensional noise signal $\boldsymbol{\eta}[n] = [\eta_1[n], \ldots, \eta_B[n]]^{\mathrm{T}}$, where $\eta_k[n] \in \mathbb{C}$ denotes the noise of receiver k. The resulting received signals are weighted with the scalar $g \in \mathbb{R}_+$, common to all receivers, to obtain the modulo inputs (see Fig. 1) which are collected in $\hat{\boldsymbol{d}}[n] = [\hat{d}_1[n], \ldots, \hat{d}_K[n]]^{\mathrm{T}} \in \mathbb{C}^B$ with $\hat{d}_k[n]$ for receiver k. The modulo operators' outputs $\hat{\boldsymbol{s}}[n] \in \mathbb{V}^B$ are mapped to the modulation alphabet $\mathbb{A}$ by the quantizer $\mathrm{Q}(\bullet)$.

The data signal $\boldsymbol{s}[n]$ is the desired signal for $\hat{\boldsymbol{s}}[n]$. Since a shift of the input of the modulo operator $\mathrm{M}(\bullet)$ by any element of $\mathbb{M}^B$ does not change the output $\hat{\boldsymbol{s}}[n]$, the transmitter has the freedom to use

$$\boldsymbol{d}[n] = \boldsymbol{s}[n] + \boldsymbol{a}[n] \tag{1}$$

with $\boldsymbol{a}[n] \in \mathbb{M}^B$ as the desired signal for the modulo inputs

$$\hat{\boldsymbol{d}}[n] = g\boldsymbol{H}\boldsymbol{T}\boldsymbol{d}[n] + g\boldsymbol{\eta}[n]. \tag{2}$$

Note that we assume that the channel $\boldsymbol{H}$ is known to the transmitter and constant over a block of N_{B} symbols. Thus, our formulation is based on time averages over N_{B} symbols.

3. VECTOR PRECODING

The VP filters and the VP perturbation signal result from following MMSE optimization

$$\begin{aligned}\{\boldsymbol{T}_{\mathrm{VP}}, g_{\mathrm{VP}}, \boldsymbol{a}_{\mathrm{VP}}[n]\} &= \underset{\{\boldsymbol{T}, g, \boldsymbol{a}[n]\}}{\operatorname{argmin}}\ \varepsilon(\boldsymbol{T}, g, \boldsymbol{a}[n]) \\ &\text{s.t.: } \sum_{n=1}^{N_{\mathrm{B}}} \mathrm{E}\left[\|\boldsymbol{y}[n]\|_2^2 \,\middle|\, \boldsymbol{s}[n]\right] = E_{\mathrm{tx}}\end{aligned} \tag{3}$$

with the MSE which is conditioned on the known data signal

$$\varepsilon(\boldsymbol{T}, g, \boldsymbol{a}[n]) = \sum_{n=1}^{N_{\mathrm{B}}} \mathrm{E}\left[\|\boldsymbol{d}[n] - \hat{\boldsymbol{d}}[n]\|_2^2 \,\middle|\, \boldsymbol{s}[n]\right]. \tag{4}$$

The filters can be readily obtained with the method of Lagrangian multipliers and can be expressed as (see [2])

$$\begin{aligned}\boldsymbol{T}_{\mathrm{VP}} &= \frac{1}{g_{\mathrm{VP}}}\boldsymbol{H}^{\mathrm{H}}\boldsymbol{\Phi}^{-1} \quad \text{and} \\ g_{\mathrm{VP}} &= \sqrt{\frac{\sum_{n=1}^{N_{\mathrm{B}}} \boldsymbol{d}_{\mathrm{VP}}^{\mathrm{H}}[n]\boldsymbol{H}\boldsymbol{H}^{\mathrm{H}}\boldsymbol{\Phi}^{-2}\boldsymbol{d}_{\mathrm{VP}}[n]}{E_{\mathrm{tx}}}}\end{aligned} \tag{5}$$

with $\boldsymbol{\Phi} = \boldsymbol{H}\boldsymbol{H}^{\mathrm{H}} + \xi\mathbf{I}_B$ and $\xi = \mathrm{tr}(\boldsymbol{R}_{\boldsymbol{\eta}})/E_{\mathrm{tx}}$. By substituting above filter solutions into the cost $\varepsilon(\boldsymbol{T}, g, \boldsymbol{a}[n])$ of (3), we find the rule for the computation of the perturbation signal

$$\boldsymbol{a}_{\mathrm{VP}}[n] = \underset{\boldsymbol{a}[n] \in \mathbb{M}^B}{\operatorname{argmin}}\ \xi(\boldsymbol{s}[n] + \boldsymbol{a}[n])^{\mathrm{H}}\boldsymbol{\Phi}^{-1}(\boldsymbol{s}[n] + \boldsymbol{a}[n]). \tag{6}$$

With the Cholesky factorization $\boldsymbol{\Phi} = \boldsymbol{U}\boldsymbol{U}^{\mathrm{H}}$, we can see that (6) is a closest point search in a lattice, i.e., we try to find the coordinates (element of $\mathbb{M}^B$) for the point of the lattice with the generator matrix $\tau\boldsymbol{U}^{-1}$ lying closest to $-\boldsymbol{U}^{-1}\boldsymbol{s}[n]$. This problem can be efficiently solved with the Schnorr-Euchner algorithm [13]. However, the problem (6) is NP-hard (e.g., [5]) and is therefore impractical.

4. TOMLINSON HARASHIMA PRECODING

The Cholesky factorization with symmetric permutation

$$\boldsymbol{\Pi}\boldsymbol{\Phi}^{-1}\boldsymbol{\Pi}^{\mathrm{T}} = \boldsymbol{L}^{\mathrm{H}}\boldsymbol{\Lambda}\boldsymbol{L} \tag{7}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix and $\boldsymbol{L}$ is unit lower triangular, enables the factorization of the precoding filter $\boldsymbol{T}$ as

$$\boldsymbol{T} = \boldsymbol{P}(\mathbf{I}_B - \boldsymbol{F})^{-1}\boldsymbol{\Pi} \tag{8}$$

with

$$\begin{aligned}\boldsymbol{P} &= g^{-1}\boldsymbol{H}^{\mathrm{H}}\boldsymbol{\Pi}^{\mathrm{T}}\boldsymbol{L}^{\mathrm{H}}\boldsymbol{\Lambda} \quad \text{and} \\ \boldsymbol{F} &= \mathbf{I}_B - \boldsymbol{L}^{-1}.\end{aligned} \tag{9}$$

The permutation matrix $\boldsymbol{\Pi} = \sum_{i=1}^{B} \boldsymbol{e}_i\boldsymbol{e}_{b_i}^{\mathrm{T}}$ fulfills $\boldsymbol{\Pi}^{-1} = \boldsymbol{\Pi}^{\mathrm{T}}$ and $b_i \in \{1, \ldots, B\}$ is the i-th element of the B-tuple

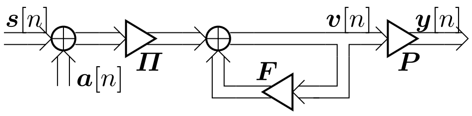

Fig. 2. Decomposition of Precoding Filter.

$\mathcal{O} = (b_1, \ldots, b_B)$ which we call precoding order. The decomposition in (8) leads to the structure of the transmitter in Fig. 2, since the feedback structure with $\boldsymbol{F}$ is an implementation of $(\mathbf{I}_B - \boldsymbol{F})^{-1}$. The symmetrically permuted Cholesky factorization (7) can also be used to rewrite (6):

$$\boldsymbol{a}_{\mathrm{VP}}[n] = \underset{\boldsymbol{a}[n]\in\mathbb{M}^B}{\operatorname{argmin}}\, \xi \left\| \boldsymbol{\Lambda}^{1/2} \boldsymbol{L}\boldsymbol{\Pi}(\boldsymbol{s}[n] + \boldsymbol{a}[n]) \right\|_2^2 .$$

We see that $\boldsymbol{L}\boldsymbol{\Pi}(\boldsymbol{s}[n] + \boldsymbol{a}[n])$ is a by-product of above lattice search. Therefore, we only need to compute the filter $\boldsymbol{P}_{\mathrm{VP}}$ and not $\boldsymbol{T}_{\mathrm{VP}}$ for VP,[1] since for the input signal $\boldsymbol{v}[n]$ of $\boldsymbol{P}_{\mathrm{VP}}$ we have (see Fig. 2)

$$\boldsymbol{v}[n] = (\mathbf{I}_B - \boldsymbol{F})^{-1}\boldsymbol{\Pi}(\boldsymbol{s}[n] + \boldsymbol{a}[n]) = \boldsymbol{L}\boldsymbol{\Pi}(\boldsymbol{s}[n] + \boldsymbol{a}[n]). \tag{10}$$

Thus, the rule for the perturbation signal (6) can be rewritten as

$$\boldsymbol{a}_{\mathrm{VP}}[n] = \underset{\boldsymbol{a}[n]\in\mathbb{M}^B}{\operatorname{argmin}}\, \xi \left\| \boldsymbol{\Lambda}^{1/2} \boldsymbol{v}[n] \right\|_2^2 . \tag{11}$$

Note that $\mathrm{E}[\|\boldsymbol{d}[n] - \hat{\boldsymbol{d}}[n]\|_2^2 \,|\, \boldsymbol{s}[n]] = \xi \boldsymbol{v}^{\mathrm{H}}[n]\boldsymbol{\Lambda}\boldsymbol{v}[n]$, since the cost of (6) is equal to the conditioned MSE of the n-th data symbol.

To avoid the high computational complexity of the lattice search (6), we can exploit the lower triangular structure of $\boldsymbol{L}$ and use the heuristic of computing the elements of $\boldsymbol{a}[n]$ successively, i.e., $a_{b_i}[n]$ is computed for given $a_{b_1}[n], \ldots, a_{b_{i-1}}[n]$ minimizing the i-th summand instead of the whole sum of the Euclidian norm:

$$a_{\mathrm{succ},b_i}[n] = \underset{a\in\mathbb{M}}{\operatorname{argmin}}\, \xi \left| \boldsymbol{e}_i^{\mathrm{T}} \boldsymbol{\Lambda}^{1/2} \boldsymbol{L}(\boldsymbol{\Pi}\boldsymbol{s}[n] + \boldsymbol{a}'[n]) \right|^2 \tag{12}$$

with $\boldsymbol{a}'[n] = [a_{\mathrm{succ},b_1}[n], \ldots, a_{\mathrm{succ},b_{i-1}}[n], a, \times, \ldots, \times]^{\mathrm{T}}$. Note that the entries '$\times$' are arbitrary due to the structure of $\boldsymbol{L}$. Let $\lambda_{i,i}$ denote the i-th diagonal element of $\boldsymbol{\Lambda}$. Then, we have that

$$\begin{aligned} \boldsymbol{e}_i^{\mathrm{T}} \boldsymbol{\Lambda}^{1/2} \boldsymbol{L} &= \sqrt{\lambda_{i,i}}\, \boldsymbol{e}_i^{\mathrm{T}} \boldsymbol{L} = \sqrt{\lambda_{i,i}} \left(\boldsymbol{e}_i^{\mathrm{T}} + \boldsymbol{e}_i^{\mathrm{T}} \boldsymbol{L} - \boldsymbol{e}_i^{\mathrm{T}} \right) \\ &= \sqrt{\lambda_{i,i}}\, \boldsymbol{e}_i^{\mathrm{T}} + \sqrt{\lambda_{i,i}}\, \boldsymbol{e}_i^{\mathrm{T}} (\mathbf{I}_B - \boldsymbol{L}^{-1}) \boldsymbol{L}. \end{aligned}$$

Therefore, above rule (12) for the successive computation of the perturbation signal $\boldsymbol{a}[n]$ can be rewritten as [see also (9) and (10)]

$$\begin{aligned} a_{\mathrm{succ},b_i} &= \underset{a\in\mathbb{M}}{\operatorname{argmin}}\, \xi \lambda_{i,i} \left| s_{b_i}[n] + a + \boldsymbol{e}_i^{\mathrm{T}} \boldsymbol{F}\boldsymbol{v}[n] \right|^2 \\ &= -\,\mathrm{Q}_{\mathbb{M}} \left(s_{b_i}[n] + \boldsymbol{e}_i^{\mathrm{T}} \boldsymbol{F}\boldsymbol{v}[n] \right). \end{aligned} \tag{13}$$

[1]To save the complexity of computing the inverse of $\boldsymbol{\Phi}$, we have to use the Cholesky factorization $\boldsymbol{\Phi} = \boldsymbol{L}'\boldsymbol{\Lambda}'\boldsymbol{L}'^{,\mathrm{H}}$ for VP. Consequently, $\boldsymbol{L}$, $\boldsymbol{\Lambda}$, and $\boldsymbol{\Pi}$ must be replaced with $\boldsymbol{L}'^{,-1}$, $\boldsymbol{\Lambda}^{-1}$, and $\mathbf{I}_B$, respectively.

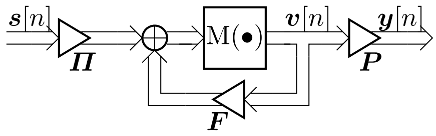

Fig. 3. Transmitter of Tomlinson Harashima Precoding.

Here, the quantizer to the closest lattice point of $\mathbb{M}$ is denoted by $\mathrm{Q}_{\mathbb{M}}(\bullet)$ which can be expressed by means of the modulo operator $\mathrm{Mod}(\bullet)$ as $\mathrm{Q}_{\mathbb{M}}(x) = x - \mathrm{Mod}(x)$ for any $x \in \mathbb{C}$. Hence, the permuted perturbation signal reads as

$$\boldsymbol{\Pi}\boldsymbol{a}_{\mathrm{succ}}[n] = \mathrm{M}\left(\boldsymbol{\Pi}\boldsymbol{s}[n] + \boldsymbol{F}\boldsymbol{v}[n]\right) - \boldsymbol{\Pi}\boldsymbol{s}[n] - \boldsymbol{F}\boldsymbol{v}[n].$$

With this result, we get for the output of the feedback loop

$$\begin{aligned} \boldsymbol{v}[n] &= \boldsymbol{\Pi}\boldsymbol{s}[n] + \boldsymbol{\Pi}\boldsymbol{a}[n] + \boldsymbol{F}\boldsymbol{v}[n] \\ &= \mathrm{M}\left(\boldsymbol{\Pi}\boldsymbol{s}[n] + \boldsymbol{F}\boldsymbol{v}[n]\right). \end{aligned}$$

We see that the addition of the successively computed perturbation signal can be replaced by a modulo operator inside the feedback loop as depicted in Fig. 3 which is the model for the transmitter of THP used in [8, 10]. As the expressions for the filters in (9) are the same as those found for THP based on the MMSE criterion in [8], we can conclude that the only difference of VP and THP is the computation of the perturbation vector. VP performs the complete lattice closest point search (6) to find the perturbation vector, whereas THP employs the heuristic of a successive computation [see (12)].

4.1. Block-Wise Precoding Order Computation

The choice of the permutation matrix $\boldsymbol{\Pi}$ which depends on the precoding order $\mathcal{O}$ strongly affects the MSE achieved by the successive computation (12) compared to (6). The straightforward approach to find the optimum precoding order for THP is to try out all $B!$ possible orders for every data vector $\boldsymbol{s}[n]$. But since the resulting computational complexity is even higher than for VP, another heuristic is usually employed for THP. First, the precoding order is computed based on an average argument, i.e., the unconditioned transmit power $\mathrm{E}[\|\boldsymbol{y}[n]\|_2^2]$ and the unconditioned MSE $\mathrm{E}[\|\boldsymbol{d}[n] - \hat{\boldsymbol{d}}[n]\|_2^2]$ are respectively used instead of the conditioned transmit power $\mathrm{E}[\|\boldsymbol{y}[n]\|_2^2 \,|\, \boldsymbol{s}[n]]$ and MSE $\mathrm{E}[\|\boldsymbol{d}[n] - \hat{\boldsymbol{d}}[n]\|_2^2 \,|\, \boldsymbol{s}[n]]$:

$$\mathrm{E}\left[\|\boldsymbol{y}[n]\|_2^2\right] = \mathrm{tr}\left(\boldsymbol{P}\boldsymbol{R}_{\boldsymbol{v}}\boldsymbol{P}^{\mathrm{H}}\right) \quad \text{and} \tag{14}$$

$$\mathrm{E}\left[\|\boldsymbol{d}[n] - \hat{\boldsymbol{d}}[n]\|_2^2\right] = \xi\, \mathrm{E}\left[\|\boldsymbol{\Lambda}^{1/2}\boldsymbol{v}[n]\|_2^2\right] = \xi\, \mathrm{tr}\left(\boldsymbol{\Lambda}\boldsymbol{R}_{\boldsymbol{v}}\right),$$

where we used the result in (11) to obtain the expression for the unconditioned MSE. Second, the approximate statistical properties of the signal $\boldsymbol{v}[n]$ are exploited (see [14, Theorem 3.1]):

$$\boldsymbol{R}_{\boldsymbol{v}} = \mathrm{diag}(\sigma_s^2, \sigma_v^2, \ldots, \sigma_v^2), \tag{15}$$

i.e., the output $\boldsymbol{v}[n]$ of the modulo operator has uncorrelated entries. Since the first symbol is not affected by the modulo operator due to $\mathbb{A} \subset \mathbb{V}$, its variance is equal to the variance σ_s^2 of the sequences $s_i[n], i = 1, \ldots, B$. The other entries of $\boldsymbol{v}[n]$ have the variance $\sigma_v^2 = \tau^2/6$ which follows from a uniform distribution over $\mathbb{V}$. The diagonal structure of $\boldsymbol{R_v}$ allows for a successive computation of the indices $b_i, i = 1, \ldots, B$, since the optimization of the i-th summand of the unconditioned MSE $\mathrm{E}[\|\boldsymbol{d}[n] - \hat{\boldsymbol{d}}[n]\|_2^2]$ with respect to b_i only depends on the i-th diagonal element $\lambda_{i,i}$ of the diagonal matrix $\boldsymbol{\Lambda}$.

The so-called *best-last* strategy of successively optimizing the precoding order relies on the observation that the performance of the data stream precoded last is crucial for a good overall performance of the precoder. This observation is supported by the argument that the interference caused by the data stream precoded last can only be suppressed linearly by the feedforward filter $\boldsymbol{P}$, whereas the parts of the interference caused by the other data streams are combatted non-linearly by the modulo feedback loop. Therefore, the index of the data stream precoded last is chosen first and the index of the data stream performing best with linear precoding is taken. The procedure is similar for the other indices; based on the already found indices of the data streams precoded later, the index is chosen leading to the best performance under the assumption that the interference to the data streams precoded later is subtracted non-linearly:

$$\mathcal{O}_{\text{best-last}}: \quad b_i = \underset{b \in \{1,\ldots,B\} \setminus \{b_{i+1},\ldots,b_B\}}{\operatorname{argmin}} \lambda_{b,b} \quad i = B, \ldots, 1. \tag{16}$$

The cost $\lambda_{b,b}$ is proportional to the MSE of the b-th data stream, since we have for the MSE matrix corresponding to the unconditioned MSE $\mathrm{E}[\|\boldsymbol{d}[n] - \hat{\boldsymbol{d}}[n]\|_2^2]$ [see (14)]:

$$\mathrm{E}\left[(\boldsymbol{d}[n] - \hat{\boldsymbol{d}}[n])(\boldsymbol{d}[n] - \hat{\boldsymbol{d}}[n])^{\mathrm{H}}\right] = \boldsymbol{\Lambda} \boldsymbol{R_v}$$

and $\boldsymbol{R_v}$ is diagonal [see (15)]. For the efficient implementation of the best-last order optimzation with the symmetrically permuted Cholesky factorization, see [8].

5. SYMBOL-WISE PRECODING ORDER OPTIMIZATION

So far we have searched for a precoding order that performs well on average for a block of many data symbols. In Fig. 4, we demonstrate how much performance can be gained by using a different precoding order for every single data symbol. We numerically simulated the transmission of blocks of $N_\mathrm{B} = 100$ data symbols over 10000 different channel realizations, where the channel coefficients were i. i. d. according to the normal distribution. We used a 16QAM symbol alphabet and assumed perfect channel knowledge at the precoder. The importance of optimizing the precoding order can be seen from the fact that THP without any sort of order optimization is far inferior to all schemes with order optimization. It

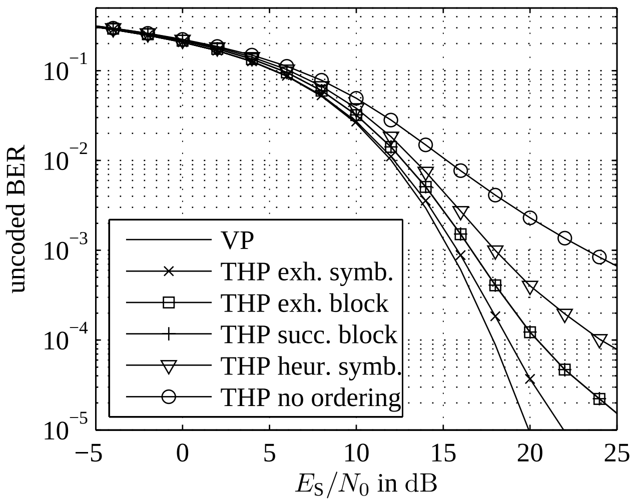

Fig. 4. Comparison of Different Approaches to Optimizing the Precoding Order for THP, $B = N = 4$, 16QAM, Uncorrelated Channels

can also be seen that optimizing the precoding order successively with the symmetrically permuted Cholesky factorization (THP succ. block, [8]) performs nearly exactly as well as when all $B!$ possible permutations are tried out and the one that minimizes the unconditioned MSE with the assumptions on the statistics of $\boldsymbol{v}[n]$ (cf. Section 4.1) is used (THP exh. block). If, however, for every symbol $n = 1, \ldots, N_\mathrm{B}$ the conditioned MSE is evaluated according to (6) for all $B!$ possible permutations and the best one is chosen for each symbol separately, we observe a considerable performance gain (THP exh. symb.). We see that a large part of the loss of THP compared to VP is due to the choice for the precoding order $\mathcal{O}$ and not due to the successive computation of the perturbation signal in (12). Clearly, the search over all possible $B!$ precoding orders is infeasible and we have to find a suboptimal strategy with less complexity to attain some of the gain possible with symbol-wise precoding order optimization.

In the following, we describe a fairly straightforward heuristic approach for determining a suitable suboptimal symbol-wise precoding order, which, however, turns out to perform worse than best-last block-wise THP. The idea is to minimize the i-th summand of the total MSE as in (12), but over the choice of the entry a of the perturbation vector *and* over the choice of the index of the data stream to be precoded $b \in \{1, \ldots, B\} \setminus \{b_1, \ldots, b_{i-1}\}$, for $i = 1, \ldots, B$. Note, however, that the successive order computation presented in the previous section follows the principle of minimizing the MSE beginning with the index b_B and proceeding backwards. Clearly, our procedure requires us to start with b_1 and to proceed forwards, as the perturbation vector by definition is determined beginning with index b_1. We therefore employ the *worst-first* strategy, which has been shown to perform nearly as well as the best-last method of (16) for block-wise order optimization (see [8]). When using the worst-first strategy, we choose the index b_i that *maximizes* the respective summand of the MSE, for $i = 1, \ldots, B$, hoping that the good indices are left over at the end, where the influence on the performance is the largest.

Our heuristic procedure for symbol-wise order optimization works as follows: in the i-th step, we determine the optimal perturbation a according to (13) for all remaining possibilities for the index b_i and choose the index that leads to the *highest* MSE. Note that $\boldsymbol{e}_i^{\mathrm{T}}\boldsymbol{F}$, as well as $\lambda_{i,i}$, which is required for computing the MSE-summand, do not depend on the indices $b_{i+1}, \ldots, b_B$, which are not yet known. This can be seen by looking at the Cholesky factorization of $\boldsymbol{\Phi}$, instead of $\boldsymbol{\Phi}^{-1}$:

$$\boldsymbol{\Pi}\boldsymbol{\Phi}\boldsymbol{\Pi}^{\mathrm{T}} = \tilde{\boldsymbol{L}}\tilde{\boldsymbol{\Lambda}}\tilde{\boldsymbol{L}}^{\mathrm{H}},$$

where $\tilde{\boldsymbol{L}} = \boldsymbol{L}^{-1}$ and $\tilde{\boldsymbol{\Lambda}} = \boldsymbol{\Lambda}^{-1}$ [cf. (7)]. The top left $i \times i$ block of $\boldsymbol{\Pi}\boldsymbol{\Phi}\boldsymbol{\Pi}^{\mathrm{T}}$ only depends on $b_1, \ldots, b_i$, which is therefore also true for the top left $i \times i$ block of $\tilde{\boldsymbol{L}}$ and $\tilde{\boldsymbol{\Lambda}}$. Since $\boldsymbol{F} = \mathbf{I}_B - \tilde{\boldsymbol{L}}$ [see (9)], $\boldsymbol{e}_i^{\mathrm{T}}\boldsymbol{F}$ is obviously not influenced by the choice of the indices $b_{i+1}, \ldots, b_B$. The computation of the symmetrically permuted Cholesky factorization of $\boldsymbol{\Phi}$ is thus performed during the successive procedure, with a new row being added to $\tilde{\boldsymbol{L}}$ after each step.

As can be seen in Fig. 4, this procedure (THP heur. symb.) does not perform very well and is even beaten by THP with a block-wise optimized precoding order. The reason for this is that the index of the last user is the one that is left over after the indices of all other users are chosen according to some criterion. Thus, the most significant contribution to the performance is not directly optimized taking into account the data symbol. Clearly, a more sophisticated approach is necessary in which more attention is paid to the last user in the precoding order.

5.1. Symbol-Wise Ordered Successive Precoding

Recall that the strategies to compute the precoding order per block are mainly based on the observation that the index of the data stream precoded last is crucial for the overall performance of the precoder. Similarly, our new *symbol-wise ordered successive precoding* (SOSP) strategy mainly gains from choosing a good index for the data stream precoded last. This is accomplished as follows.

The optimal index of the data stream precoded last can be the index of any data stream, i.e., $b_B \in \{1, \ldots, B\}$. Therefore, we must test all B values for b_B and choose the index with the best MSE [cost of (6) or (11)]. Since the strategy of successively choosing the precoding order for each symbol separately as described above leads to poor results, we rely on the block-wise best-last strategy to choose the other indices, i.e., for every value of b_B, the other indices $b_1, \ldots, b_{B-1}$ are found successively starting with b_{B-1} following the best-last

1: $\boldsymbol{\Phi}^{-1} \leftarrow (\boldsymbol{H}\boldsymbol{H}^{\mathrm{H}} + \xi\mathbf{I}_B)^{-1}$
for $k = 1, \ldots, B$
$\boldsymbol{\Lambda}_k \leftarrow \mathbf{0}_{B\times B}$, $\boldsymbol{\Pi}_k \leftarrow \mathbf{I}_B$, $\boldsymbol{A} \leftarrow \boldsymbol{\Phi}^{-1}$
4:    $q \leftarrow k$
5:    for $i = B, \ldots, 2$
$\boldsymbol{\Pi} \leftarrow \mathbf{I}_B$, whose i-th and q-th rows are exchanged
$\boldsymbol{\Pi}_k \leftarrow \boldsymbol{\Pi}\boldsymbol{\Pi}_k$
$\boldsymbol{A} \leftarrow \boldsymbol{\Pi}\boldsymbol{A}\boldsymbol{\Pi}^{\mathrm{T}}$
$\boldsymbol{\Lambda}_k(i,i) \leftarrow \boldsymbol{A}(i,i)$
$\boldsymbol{A}(1:i,i) \leftarrow \boldsymbol{A}(1:i,i)/\boldsymbol{\Lambda}_k(i,i)$
$\boldsymbol{A}(1:i-1,1:i-1) \leftarrow \boldsymbol{A}(1:i-1,1:i-1)$
$-\boldsymbol{A}(1:i-1,i)\boldsymbol{A}(1:i-1,i)^{\mathrm{H}}\boldsymbol{\Lambda}_k(i,i)$
12:     $q \leftarrow \mathrm{argmin}_{b\in\{1,\ldots,i-1\}}\, \boldsymbol{A}(b,b)$
$\boldsymbol{\Lambda}_k(1,1) \leftarrow \boldsymbol{A}(1,1)$
$\boldsymbol{A}(1,1) \leftarrow 1$
$\boldsymbol{L}_k \leftarrow$ lower triangular part of $\boldsymbol{A}$
16: $\boldsymbol{T}' \leftarrow \boldsymbol{H}^{\mathrm{H}}\boldsymbol{\Phi}^{-1}$

Table 1. Pseudo Code for Precoding Order and Filter Computation.

rule in (16). For $k \in \{1, \ldots, B\}$,

$$\mathcal{O}_k: \quad b_B = k \quad \text{and} \qquad b_i = \underset{b\in\{1,\ldots,B\}\setminus\{b_{i+1},\ldots,b_B\}}{\mathrm{argmin}} \lambda_{b,b} \quad i = B-1, \ldots, 1. \tag{17}$$

The computation of above precoding orders $\mathcal{O}_k, k = 1, \ldots, B$, is independent of the data signal $\boldsymbol{s}[n]$. Therefore, the B precoding orders and the respective Cholesky factorizations (7) only have to be computed once per data block. The pseudo code is given in Table 1.

The core (lines 5–12) of the algorithm is the standard loop for the computation of a symmetrically permuted Cholesky factorization (see [15]), where the elements of the diagonal matrix $\boldsymbol{\Lambda}_k$ are successively minimized [8]. Since we need the factorizations (7) corresponding to the precoding orders $\mathcal{O}_k, k = 1, \ldots, B$, the index of the last data stream is set to k in line 4. The filter $\boldsymbol{T}'$ computed in line 16 follows from (5), when dropping the normalization with g.

For every of the precoding orders $\mathcal{O}_k, k = 1, \ldots, B$, we find the Cholesky factors $\boldsymbol{L}_k$ and $\boldsymbol{\Lambda}_k$ with Table 1. When precoding the data signal $\boldsymbol{s}[n]$, we can compute the perturbation signal $\boldsymbol{a}[n]$ according to (12) for every $k = 1, \ldots, B$:

$$\begin{aligned} a_{k,b_i}[n] &= \underset{a\in\mathbb{M}}{\mathrm{argmin}}\, \xi\lambda_{i,i} \left| \boldsymbol{\ell}_{k,i}^{\mathrm{T}}(\boldsymbol{\Pi}_k\boldsymbol{s}[n] + \boldsymbol{a}'_k[n]) \right|^2 \\ &= \underset{a\in\mathbb{M}}{\mathrm{argmin}} \left| \boldsymbol{\ell}_{k,i}^{\mathrm{T}}\boldsymbol{s}_k^{(i)}[n] + a) \right|^2 \\ &= -\,\mathrm{Q}_{\mathbb{M}}\left(\boldsymbol{\ell}_{k,i}^{\mathrm{T}}\boldsymbol{s}_k^{(i)}[n] \right). \end{aligned} \tag{18}$$

Here, $\boldsymbol{s}_k^{(i)}[n] = \boldsymbol{\Pi}_k\boldsymbol{s}[n] + [a_{k,b_1}, \ldots, a_{k,b_{i-1}}, 0, \ldots, 0]^{\mathrm{T}}$ and $\boldsymbol{\ell}_{k,i}^{\mathrm{T}}$ is the i-th row of $\boldsymbol{L}_k$. Since $\boldsymbol{v}_k[n] = \boldsymbol{L}_k\boldsymbol{\Pi}_k(\boldsymbol{s}[n] +$

1: for $n = 1, \ldots, N_\mathrm{B}$
2: $\quad \varepsilon \leftarrow \infty$
$\quad$ for $k = 1, \ldots, B$
$\qquad \boldsymbol{s}_k \leftarrow \boldsymbol{\Pi}_k \boldsymbol{s}[n], \boldsymbol{v}_k \leftarrow \boldsymbol{s}_k$
$\qquad$ for $i = 2, \ldots, B$
6: $\qquad\quad \alpha \leftarrow \boldsymbol{L}_k(i,:)\boldsymbol{s}_k$
7: $\qquad\quad a_{k,i} \leftarrow -\tau \left\lfloor \frac{\mathrm{Re}(\alpha)}{\tau} + \frac{1}{2} \right\rfloor - \mathrm{j}\, \tau \left\lfloor \frac{\mathrm{Im}(\alpha)}{\tau} + \frac{1}{2} \right\rfloor$
8: $\qquad\quad \boldsymbol{s}_k(i) \leftarrow \boldsymbol{s}_k(i) + a_{k,i}$
9: $\qquad\quad \boldsymbol{v}_k(i) \leftarrow \alpha + a_{k,i}$
10: $\qquad \varepsilon_k \leftarrow \xi \boldsymbol{v}_k^\mathrm{H} \boldsymbol{\Lambda}_k \boldsymbol{v}_k$
$\qquad$ if $\varepsilon_k < \varepsilon$
12: $\qquad\quad \varepsilon \leftarrow \varepsilon_k, k_\mathrm{SOSP} \leftarrow k$
13: $\quad \boldsymbol{y}'[n] \leftarrow \boldsymbol{T}' \boldsymbol{\Pi}_{k_\mathrm{SOSP}} \boldsymbol{s}_{k_\mathrm{SOSP}}$
14: $g \leftarrow \sqrt{\sum_{n=1}^{N_\mathrm{B}} \lVert \boldsymbol{y}'[n] \rVert_2^2 / E_\mathrm{tx}}$
for $n = 1, \ldots, N_\mathrm{B}$
$\quad \boldsymbol{y}[n] \leftarrow \boldsymbol{y}'[n]/g$

Table 2. Symbol-Wise Precoding Order Optimization with SOSP

$\boldsymbol{a}_k[n]$), we obtain for the i-th element of $\boldsymbol{v}_k[n]$:

$$\begin{aligned} v_{k,i}[n] &= \boldsymbol{\ell}_{k,i}^\mathrm{T} \boldsymbol{s}_k^{(i)}[n] + a_{k,b_i}[n] \\ &= \mathrm{M}\left(\boldsymbol{\ell}_{k,i}^\mathrm{T} \boldsymbol{s}_k^{(i)}[n]\right). \end{aligned} \tag{19}$$

Remember that $\mathrm{M}(x) = x - \mathrm{Q}_\mathbb{M}(x)$. Hence, the i-th entry $a_{k,b_i}[n]$ of the permuted perturbation signal $\boldsymbol{\Pi}_k \boldsymbol{a}_k[n]$ is found as a by-product when computing the i-th entry $v_{k,i}[n]$ of the signal $\boldsymbol{v}_k[n]$.

The SOSP strategy chooses the precoding order $\mathcal{O}_{k_\mathrm{SOSP}}$ leading to the minimum MSE [cf. (11)]:

$$k_\mathrm{SOSP} = \underset{k \in \{1,\ldots,B\}}{\operatorname{argmin}} \; \xi \boldsymbol{v}_k^\mathrm{H}[n] \boldsymbol{\Lambda}_k \boldsymbol{v}_k[n]. \tag{20}$$

The pseudo code for precoding a block of N_B symbols is given in Table 2. For every of the N_B symbols, we test the B different precoding orders. If a better MSE is found (the MSE is initialized with infinity in line 2), the index is updated in line 12. As obtained in (18), the i-th entry of the permuted perturbation signal is found in line 7 by quantizing $\boldsymbol{\ell}_{k,i}^\mathrm{T} \boldsymbol{s}_k^{(i)}[n]$. With line 8, we get $\boldsymbol{s}_k^{(i+1)}[n]$ from $\boldsymbol{s}_k^{(i)}[n]$ and the modulo operation of (19) is computed in line 9. Finally, the normalization g of the transmit signal to fulfill the transmit power constraint is found via (5) in line 14.

6. PERFORMANCE COMPARISON

6.1. Complexity of the Precoding Schemes

For the complexity analysis of the discussed precoding schemes, we count each complex addition, complex multiplication, and division as one *floating point operation* (FLOP).

operation	complexity order
Gram $\boldsymbol{H}\boldsymbol{H}^\mathrm{H}$	NB^2
Cholesky factorization of $\boldsymbol{\Phi}$ or $\boldsymbol{\Phi}^{-1}$	$\frac{1}{3}B^3$
inversion of $\boldsymbol{\Phi}$	B^3
multiplication $\boldsymbol{H}^\mathrm{H}\boldsymbol{\Pi}^\mathrm{T}\boldsymbol{L}$	NB^2
inversion & multiplication $\boldsymbol{H}^\mathrm{H}\boldsymbol{L}'^{,-1}$	NB^2
multiplication $\boldsymbol{P}\boldsymbol{v}[n]$	$2NB$
multiplication $\boldsymbol{L}\boldsymbol{\Pi}\boldsymbol{s}[n]$	B^2

Table 3. Order of Complexity of Basic Operations

	filter computation	precoding operation
linear precoding	$\frac{10}{3}B^3$	$2B^2$
best-last THP	$\frac{10}{3}B^3$	$3B^2$
SOSP	$\frac{1}{3}B^4 + \frac{9}{2}B^3$	$B^3 + 7B^2$
VP	$\frac{8}{3}B^3$	non-polynomial

Table 4. Order of Complexity of Discussed Precoding Approaches

Table 3 shows the complexity orders of the FLOP counts for the basic operations of the precoding schemes.

Analyzing the precoding schemes using the complexity orders from Table 3 under the assumption of $N = B$ for better comparability yields the complexity orders given in Table 4. Note that for VP, we exploit the decomposition depicted in Fig. 2, which saves us from having to explicitly compute $\boldsymbol{T}_\mathrm{VP}$ and thus makes the filter computation more efficient than for the linear precoder. Also note that with best-last THP, it is not necessary to compute the filter $\boldsymbol{F}_\text{best-last}$, since (12) delivers $\boldsymbol{v}[n]$ as a by-product.

The order of complexity expressions in Table 4 show that the proposed SOSP has a quartic complexity order for the filter computation (all other schemes have cubic complexity) and a cubic complexity order for the precoding operation which is one order of magnitude larger than for linear precoding and the two other THP schemes. However, SOSP has polynomial complexity contrary to VP. Therefore, it is still a good alternative to VP in terms of complexity.

6.2. Lattice Reduction

Similar to standard THP, the proposed SOSP strategy can be further enhanced by inserting a *lattice basis reduction* step [16, 17]. Recall that the optimum perturbation vector is the solution to a closest point search (6) in a lattice with the generator matrix $\tau\boldsymbol{U}^{-1}$. If the generator matrix is multiplied from the right with a unimodular matrix $\boldsymbol{M}$, i.e., a matrix that fulfills $|\det(\boldsymbol{M})| = 1$, the result is a generator matrix, or *basis*, of the same lattice [11, 5]. For the complete lattice search of VP, the choice of the lattice basis makes no difference. The suboptimal successive rule (12), on the other hand, yields different results for different generator matrices of the same lattice. In particular, if the lattice basis is orthogonal, the successive rule leads to the optimal perturbation vector. It is therefore desirable for successive precoding to find a unimodular matrix $\boldsymbol{M}$ that leads to an equivalent generator matrix with 'close to orthogonal' columns. The *Lenstra-Lenstra-Lovász* (LLL) algorithm [18] finds such a 'reduced' basis in fourth-order polynomial time [5], and it has been shown that successive precoding with an LLL-reduced basis achieves full diversity order, in contrast to THP [19]. The application of the lattice reduction algorithm adds an $\mathrm{O}(B^4)$ complexity term to the filter computation cost in Table 4. For the implementation of lattice reduction aided precoding, see [16, 17].

6.3. Simulation Results

For the numerical simulation results of Figs. 5 and 6, we used the real valued representation of THP (cf. [16]), in order to ensure a fair comparison with the lattice reduction aided methods, which require a real valued representation. (Note that the performance of THP in Fig. 5 is therefore superior to that in Fig. 4, albeit at a slightly higher complexity.) Otherwise, the simulation parameters are the same as in Section 5.

We compare THP with block-wise precoding order computation using the symmetrically permuted Cholesky factorization (THP) and THP with symbol-wise precoding order optimization following the proposed SOSP strategy. Furthermore, we simulated both methods using a reduced lattice basis found with the LLL-Algorithm (LR-THP, LR-SOSP). In Fig. 5, with $B = N = 4$ users and transmit antennas, our new strategy combined with lattice reduction is very close to the optimum (VP). It is also apparent that the application of the lattice reduction algorithm yields a considerable gain, as the slope of the graphs for successive precoding without lattice reduction decreases visibly due to the lower diversity order. Nonetheless, the advantage of the proposed SOSP method is clearly visible.

For $B = N = 10$ (Fig. 6), the gain through symbol-wise order optimization is much larger. Our method comes quite close to the optimum performance and clearly outperforms THP with block-wise order optimization, regardless of whether lattice reduction is applied. As the lower diversity order of successive precoding without lattice reduction is not visible in the depicted SNR region, the gain through lattice reduction is not very large.

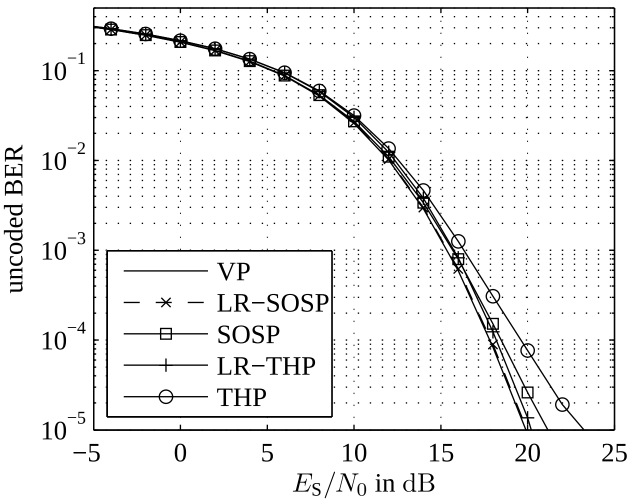

Fig. 5. Performance of Proposed Scheme, $B = N = 4$, 16QAM, Uncorrelated Channels

7. CONCLUSION

After discussing the MSE-optimal precoder for decentralized receivers equipped with a modulo operator, we showed how by employing a simple heuristic we arrive at the well known successive precoder. The structure of the successive precoder, or THP, allows us to freely choose a precoding order, in order to further minimize the MSE. While in all previous work on THP the precoding order is kept constant for the whole block of data symbols, we showed that making use of the knowledge of the data symbols at the precoder in order to choose a different order at each time instance can significantly improve the performance of THP. Furthermore, we proposed a technique for finding a good precoding order for a given vector of data symbols. Our SOSP strategy is based on trying out B different orders, each with a different user to be precoded last, and comparing the resulting MSEs. The complexity of this scheme is one order higher than that of conventional THP, but still polynomial. The performance, on the other hand, is very near to that of the optimal precoder that has exponential complexity. Our strategy can furthermore be combined with well known lattice reduction techniques, in order to further improve performance in some scenarios.

8. REFERENCES

[1] B. M. Hochwald, C. B. Peel, and A. L. Swindlehurst, "A Vector-Perturbation Technique for Near-Capacity Multi-Antenna Multi-User Communication—Part II: Perturbation," *IEEE Transactions on Communications*, vol. 53, no. 3, pp. 195–202, March 2005.

[2] D. A. Schmidt, M. Joham, and W. Utschick, "Minimum Mean Square Error Vector Precoding," *Accepted for publication in European Transactions on Telecommunications*, 2006.

[3] B. R. Vojčić and W. M. Jang, "Transmitter Precoding in Synchronous Multiuser Communications," *IEEE Transactions on Communications*, vol. 46, no. 10, pp. 1346–1355, October 1998.

[4] M. Joham, W. Utschick, and J. A. Nossek, "Linear Transmit Processing in MIMO Communications Systems," *IEEE Transactions on Signal Processing*, vol. 53, no. 8, pp. 2700–2712, August 2005.

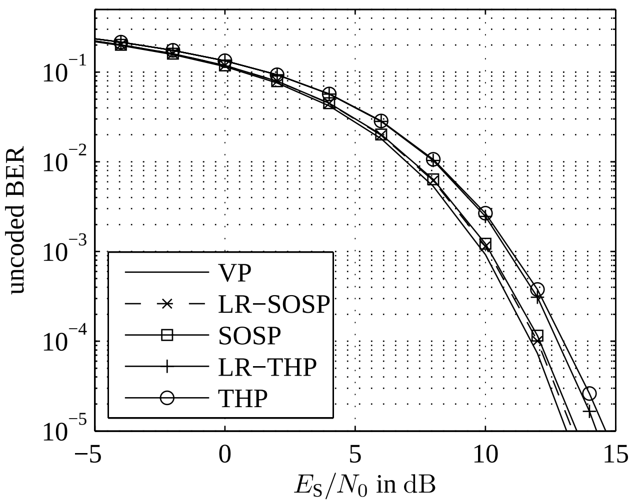

Fig. 6. Performance of Proposed Scheme, $B = N = 10$, 16QAM, Uncorrelated Channels

[5] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer, 1993.

[6] G. Ginis and J. M. Cioffi, "A Multi-user Precoding Scheme achieving Crosstalk Cancellation with Application to DSL Systems," in *Proc. Asilomar Conference on Signals, Systems, and Computers*, October 2000, vol. 2, pp. 1627–1631.

[7] C. Windpassinger, R. F. H. Fischer, T. Vencel, and J. B. Huber, "Precoding in Multiantenna and Multiuser Communications," *IEEE Transactions on Wireless Communications*, vol. 3, no. 4, pp. 1305–1316, July 2004.

[8] K. Kusume, M. Joham, W. Utschick, and G. Bauch, "Efficient Tomlinson-Harashima Precoding for Spatial Multiplexing on Flat MIMO Channel," in *Proc. ICC 2005*, May 2005, vol. 3, pp. 2021–2025.

[9] J. Liu and W. A. Krzymień, "A Novel Nonlinear Precoding Algorithm for the Downlink of Multiple Antenna Multi-User Systems," in *Proc. VTC 2005 Spring*, May 2005, vol. 2, pp. 887–891.

[10] M. Joham and W. Utschick, "Ordered Spatial Tomlinson Harashima Precoding," in *Smart Antennas — State-of-the-Art*, T. Kaiser, A. Bourdoux, H. Boche, J. R. Fonollosa, J. Bach Andersen, and W. Utschick, Eds., vol. 3 of *EURASIP Book Series on Signal Processing and Communications*, chapter III. Transmitter, pp. 401–422. EURASIP, Hindawi Publishing Corporation, 2006.

[11] J. H. Conway and N. J. A. Sloane, *Sphere Packings, Lattices and Groups*, Springer, 1999.

[12] M. Joham, D. A. Schmidt, J. Brehmer, and W. Utschick, "Finite-Length MMSE Tomlinson-Harashima Precoding for Frequency Selective Vector Channels," *Accepted for publication in IEEE Transactions on Signal Processing*, 2006.

[13] C. P. Schnorr and M. Euchner, "Lattice Basis Reduction: Improved Practical Algorithms and Solving Subset Sum Problems," *Mathematical Programming*, vol. 66, pp. 181–199, August 1994.

[14] R. F. H. Fischer, *Precoding and Signal Shaping for Digital Transmission*, John Wiley & Sons, 2002.

[15] G. Golub and C. V. Loan, *Matrix Computations*, Johns Hopkins University Press, 1996.

[16] C. Windpassinger, R. F. H. Fischer, and J. B. Huber, "Lattice-Reduction-Aided Broadcast Precoding," *IEEE Transactions on Communications*, vol. 52, no. 12, pp. 2057–2060, December 2004.

[17] G. Psaltopoulos, M. Joham, and W. Utschick, "Comparison of Lattice Search Techniques for Nonlinear Precoding," in *Proc. ITG/IEEE WSA 2006*, March 2006.

[18] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász, "Factoring Polynomials with Rational Coefficients," *Mathematische Annalen*, vol. 261, pp. 515–534, 1982.

[19] M. Taherzadeh, A. Mobasher, and A. K. Khandani, "LLL Lattice-Basis Reduction Achieves the Maximum Diversity in MIMO Systems," in *Proc. ISIT 2005*, September 2005, pp. 1300–1304.