Lattice-Reduction Aided HNN for Vector Precoding

Vesna Gardašević, Ralf R. Müller, Daniel J. Ryan, Lars Lundheim and Geir E. Øien
Department of Electronics and Telecommunications
The Norwegian University of Science and Technology
7491 Trondheim, Norway
Email: {vesna.gardasevic, mueller, daniel.ryan, lundheim, oien}@iet.ntnu.no

Abstract—In this paper we propose a modification of the Hopfield neural networks for vector precoding, based on Lenstra, Lenstra, and Lovász lattice basis reduction. This precoding algorithm controls the energy penalty for system loads $\alpha = K/N$ close to 1, with $N$ and $K$ denoting the number of transmit and receive antennas, respectively. Simulation results for the average transmit energy as a function of $\alpha$ show that our algorithm improves performance within the range $0.9 \leq \alpha \leq 1$, between $0.1$ dB and $2.6$ dB in comparison to standard HNN precoding. The proposed algorithm performs close to the sphere encoder (SE) while requiring much lower complexity, and thus, can be applied as an efficient suboptimal precoding method.

I. INTRODUCTION

We consider the application of precoding algorithms in a broadcast MIMO system, typically consisting of a transmitter with $N$ antennas and $K$ receivers. The aim of these methods is to improve the system performance, and to provide simplified receiver design. Precoding algorithms can be based on linear or a combination of linear and nonlinear transformations. The main advantage of linear precoding (e.g., [1], [2]) is in its low-complexity implementation. However, this method does not provide good performance unless the load is very small.

In order to obtain practical precoding algorithms providing higher data rates, methods that combine linear and nonlinear algorithms have been proposed (e.g., [3]). One approach in nonlinear precoding is to apply Tomlinson-Harashima precoding (THP) based on the zero forcing (ZF THP) [3] or the minimum mean square error (MMSE) (MMSE THP) (e.g., [4]) criterion. Further performance improvement is achieved by vector perturbation [5], a popular precoding method that is a generalization of the THP concept. In vector perturbation, the transmit signal is composed of a sum of the data symbols and a complex perturbation vector. The perturbation vector is chosen such that the transmit power is minimized. Numerical results show that vector perturbation achieves significant performance enhancement for all signal-to-noise ratio (SNR) regions. However, state-of-the-art vector perturbation employs a sphere encoder (SE) [6] with exponential expected complexity [7] for the search of the perturbation vector.

Reductions in the computational complexity of vector precoding have been provided by algorithms based on lattice-basis reduction (LR) (e.g., [8], [9]). Precoding based on LR algorithms that employ the polynomial-time Lenstra, Lenstra, and Lovász (LLL) algorithm [10] have been investigated in e.g., [9]. The LLL algorithm calculates an alternative basis with less correlated, short vectors, such that detection and precoding are improved due to less power enhancement. Another practical algorithm is convex relaxation (CR) precoding proposed in [11]. CR precoding is an algorithm of polynomial complexity, where the minimization of the transmit power is formulated as a convex optimization problem.

In this paper we extend our results on HNN precoding [12] by implementing a LLL-LR-based modification of the HNN precoding algorithm. Sole HNN precoding as investigated in [12] provides performance close to the SE precoding within the load $0 < \alpha < 0.6$, where the load $\alpha = K/N$ is the ratio of the number of receive antennas $K$ to the number of transmit antennas $N$. It has been shown that sole HNN exhibits a degradation for loads close to 1. We show that the LLL-LR HNN obtains improvements compared to the performances of sole HNN precoding within the system load $0.9 \leq \alpha \leq 1$. LLL-LR improves performance for HNNs, while for the SE it only moderates complexity, still keeping it exponential. For example, for $K = 4$, the LLL-LR HNN achieves performance very close to the performance of the SE, while keeping essentially the same complexity as linear precoding.

We investigate the performances of LLL-LR HNN by simulations. Numerical results for the LLL-LR HNN will be given in terms of the average transmit power as a function of the system load $\alpha$, for a MIMO system with $K = 4$, $K = 8$, and $K = 16$ receive antennas. We compare numerical results with the performances of: the sole HNN for vector precoding for the same number of receive antennas, the convex relaxation (CR) [11] for $K \rightarrow \infty$, the SE where $K \rightarrow \infty$ [13], and with numerical results for the SE lattice precoding, where each information bit is represented by $L = 2$ redundant symbols.

Notation: Matrices and vectors are denoted by uppercase boldface and lowercase boldface letters, respectively and scalars by ordinary letters. $(\cdot)^{-1}$, $(\cdot)^T$, $(\cdot)^*$, and $(\cdot)^\dagger$ denote the inverse, the transpose operator, the pseudo-inverse and the conjugate transpose, respectively. The Euclidean norm is denoted by $\| \cdot \|$, inner product of two vectors by $\langle \cdot , \cdot \rangle$, rounding to the nearest integer value by $\lfloor \cdot \rfloor$, and absolute value by $| \cdot |$. The $\Re$ and $\Im$ prefix denote the real and imaginary parts of a complex matrix and vector.

II. SYSTEM MODEL

We consider a Gaussian broadcast MIMO channel with a single transmitter with $N$ antennas, and $K$ non-cooperative receivers. Each receiver is equipped with a single antenna. In
our scenario the number of transmit antennas is greater than or equal to the number of receive antennas, $K \leq N$. Perfect channel state information (CSI) is available at the transmitter. The MIMO system is modeled as

$$\tilde{r} = \tilde{H} \tilde{t} + \tilde{n}$$  \hspace{1cm} (1)$$

where the transmit vector is denoted by $\tilde{t} = [\tilde{r}_1, \tilde{r}_2, \cdots, \tilde{r}_K]^T$, the receive vector is denoted by $\tilde{r} = [\tilde{r}_1, \tilde{r}_2, \cdots, \tilde{r}_K]^T$, and elements of the white Gaussian noise vector $\tilde{n} = [\tilde{n}_1, \tilde{n}_2, \cdots, \tilde{n}_K]^T$ are complex-valued Gaussian random variables with zero-mean and unit variance. The vector $\tilde{s} = [\tilde{s}_1, \tilde{s}_2, \cdots, \tilde{s}_K]^T$ contains quaternary phase-shift keying (QPSK) data symbols, where the $\tilde{s}_i$ entries are from the set $\tilde{S} = \{1 + j, -1 + j, 1 - j, -1 - j\}$, for $i = 1, 2, \cdots, K$. The elements of the $K \times N$ channel matrix $\tilde{H}$ are independent and identically distributed (i.i.d.) complex-valued Gaussian random variables with zero-mean and unit variance.

The complex-valued model of the system (1) can be represented as a corresponding $2K$ real-valued system. The real and imaginary parts of the complex-valued matrix and vectors are included into the system model; thus the $2K \times 2N$ real-valued MIMO system (1) is reformulated as

$$\begin{bmatrix} \Re F \\ \Im F \end{bmatrix} = \begin{bmatrix} \Re \tilde{H} & -\Im \tilde{H} \\ \Im \tilde{H} & \Re \tilde{H} \end{bmatrix} \begin{bmatrix} \Re \tilde{t} \\ \Im \tilde{t} \end{bmatrix} + \begin{bmatrix} \Re \tilde{n} \\ \Im \tilde{n} \end{bmatrix}$$

The real equivalent of the data symbol vector $\tilde{s}$, denoted by $s$, is mapped to the vector $x = [x_1, x_2, \cdots, x_{2K}]^T$, obtained as a solution of a transmit energy minimization to be performed. The entries of the real equivalent data vector $s = [s_1, s_2, \cdots, s_{2K}]^T$ are binary phase-shift keying (BPSK) modulated symbols that belong to the set $\tilde{S} = \{ \pm 1 \}$. For each BPSK symbol, we allow for a redundant representation consisting of two symbols, such that the relaxed alphabet sets are given by: $B_1 = \{ -1, 3 \}$ and $B_2 = \{-1, 3\}$. In the linear precoding step, the transmit vector $t$ is premultiplied with the pseudo-inverse of the channel matrix $H$, denoted by the matrix $T$, i.e.,

$$t = Tx$$  \hspace{1cm} (2)$$

where $T$ is given as

$$T = H^+ = H^H(HH^H)^{-1}$$  \hspace{1cm} (3)$$

and the $2K$ real-valued vector $x$ has been obtained as a solution of the optimization problem for the transmit energy minimization.

We start by formulating the expression for the minimization of the transmit energy for the system (1), and next we consider the optimization problem of the transmit energy minimization. The optimal vector $x$ is obtained as a solution to the optimization problem:

$$x^* = \arg \min_{x \in B_1 \times \cdots \times B_{2K}} \|Tx\|^2 = \arg \min_{x \in B_1 \times \cdots \times B_{2K}} x^H(\tilde{H}^H)^{-1}x$$  \hspace{1cm} (4)$$

This optimization problem belongs to the class of nonconvex optimization problems in a high dimensional lattice, and it is known to be NP-hard. In order to provide a lower complexity suboptimal solution for the optimization problem (5), in the next section, we proceed with the derivation of the LLL-LR HNN precoding algorithm for the selection of the optimal vector $x$.

A. Lattice Basis Reduction Algorithm

The search for a less correlated basis with short vectors in a lattice set can be carried out by lattice reduction algorithms [14], for example the Minkowski or LLL reduction algorithms. In order to proceed with the implementation of the precoding algorithm based on lattice basis reduction, definitions [14] and a brief introduction on general lattices will be given in this subsection.

A lattice or $\mathbb{Z}$-module, denoted by $L = \sum_{i=1}^d \mathbb{Z}b_i = \{ \sum_{i=1}^d \lambda_i b_i : \lambda_1, \cdots, \lambda_d \in \mathbb{Z} \}$ is a set of all integer combinations of the vectors $b_1, b_2, \cdots, b_d \in \mathbb{R}^n$, where $n \geq d$, and $d$ is the lattice dimension. If the vectors $(b_1, b_2, \cdots, b_d)$ are linearly independent, they constitute a basis of the lattice. Infinitely many different bases can be constructed for $d \geq 2$. Let $B$ be the matrix $B = [b_1, b_2, \cdots, b_d]$, consisting of the column vectors $b_1, b_2, \cdots, b_d$. The lattice basis obtained by the LLL algorithm is not necessarily the best one, but it is only by a certain factor worse than the best basis.

Applying the LLL algorithm to the columns of $B$, a new reduced basis $L(B)$, with basis vectors $b_1, b_2, \cdots, b_d$ is obtained. Those two lattice bases are related to each other via an unimodular transformation, that is $B_{red} = BU$, where $U$ is a unimodular matrix, i.e. a matrix consisting of integers, with normalized determinant $|\det(U)| = 1$.

Let the LR of the pseudo-inverse $H^+$ be denoted by $H_{red}^+ = H^+U$, where $U$ is a unimodular matrix. Since the basis $H_{red}^+$ has better properties in terms of basis vector length and orthogonality, less average transmitted energy will be required. We will at the moment neglect the noise in (1) and consider the received vector given by $HH^HUX$. Given that the data vector is represented by the shifted lattice $2Z + 1$, the $2K$-dimensional received vector $r$ should be represented by a shifted lattice as well, in order to provide interference-free transmission.

Let $1$ be the 2K vector consisting of ones. For any vector $x \in (2Z + 1)^{2K}$, we have $UX \in (2Z)^{2K} + U1$. Thus, an unimodular matrix acting on a shifted lattice does not distort the lattice, but only shifts it. Assuming this constant shift $U1$ to be known to the receiver, it can easily be compensated for.

III. HOPFIELD NEURAL NETWORKS

The HNN [15] is modeled as a system consisting of interconnected signal processing units -- neurons, an external threshold value $\theta_i$, and an activation function. Each connection between two neurons is determined by a weight, and the set of all weights in the HNN is represented by a weight matrix $W$. The choice of the weight matrix $W$ depends on the underlying system that is modeled by the HNN. An input signal at a
neuron is usually a binary valued signal \{1, -1\} or \{1, 0\}. A task of an external threshold value \(\theta_i\) is to control an amplitude of the signal at the input of the activation function.

In the HNN, the amplitude of the sum of a linear combination of the weighted input states and the threshold \(\theta_i\) is limited by an activation function. The HNN employs feedback, and the output of the activation function is sent at the input of the neurons. The activation function can be implemented by various functions, for example: hard limiter (threshold) transfer function, hyperbolic tangent (tanh), sigmoid and other functions.

The mathematical model of discrete HNN [15] is given in the form

\[
v^{(l+1)}_j = f \left( \sum_{i=1}^{K} w_{ji} v^{(l)}_i + \theta_j \right)
\]

where \(l = 1, 2, \ldots\), denotes the number of iterations run by the HNN, \(i = 1, 2, \ldots, K\), \(j = 1, 2, \ldots, K\), the network states are denoted by \(v_{ji}\), \(w_{ji}\) are the assigned weights between neurons \(j\) and \(i\), \(f(\cdot)\) is an activation function, and \(\theta_j\) is a threshold value.

The energy function of the HNN is described by

\[
E = \frac{1}{2} \sum_{i=1}^{K} \sum_{j=1}^{K} w_{ji} v_{i} v_{j} + \sum_{i=1}^{K} v_{i} \theta_{i} \tag{6}
\]

The algorithm of the HNN based on the lattice basis reduction for vector precoding is shown in Table I.

The LLL-LR HNN precoding algorithm works as follows: The LLL algorithm is applied to the pseudo-inverse \(H^+\) of the channel matrix, and \(H^+_{\text{red}} = H^+ U\) is obtained. The transmit vector is given by \(t = H^+_{\text{red}} x\), and the optimization problem is formulated as minimizing \(|x^T (H^+_{\text{red}})^\dagger H^+_{\text{red}} (H^+_{\text{red}})^\dagger x|\) over the constraint set \(x \in B_1 \times \cdots \times B_{2^K}\).

The LLL algorithm is shown in Table I.

### Table I

<table>
<thead>
<tr>
<th>Algorithm 1: LLL Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> Lattice basis ((b_1, b_2, \ldots, b_d))</td>
</tr>
<tr>
<td>1: Gram-Schmidt orthogonalization step</td>
</tr>
<tr>
<td>2: Compute the Gram-Schmidt orthogonal basis:</td>
</tr>
<tr>
<td>(\tilde{b}_1, \tilde{b}_2, \ldots, \tilde{b}_d)</td>
</tr>
<tr>
<td>4: Size Reduction step</td>
</tr>
<tr>
<td>5: for (2 \leq i \leq d) do</td>
</tr>
<tr>
<td>6: for (i - 1 \leq j \leq d) do</td>
</tr>
<tr>
<td>7: Define: (\mu_{i,j} =</td>
</tr>
<tr>
<td>8: (b_i \leftarrow b_i - [\mu_{i,j}] b_j)</td>
</tr>
<tr>
<td>9: Swap the vectors:</td>
</tr>
<tr>
<td>10: if (</td>
</tr>
<tr>
<td>11: then</td>
</tr>
<tr>
<td>12: Swap the vectors (b_{i-1}) and (b_i)</td>
</tr>
<tr>
<td>13: (\tilde{b}<em>i \leftarrow \tilde{b}</em>{i-1})</td>
</tr>
<tr>
<td>14: (i \leftarrow i - 1)</td>
</tr>
<tr>
<td>15: else</td>
</tr>
<tr>
<td>16: (i \leftarrow i + 1)</td>
</tr>
<tr>
<td>17: end</td>
</tr>
<tr>
<td>18: Go to Gram-Schmidt orthogonalization step</td>
</tr>
<tr>
<td>19: end</td>
</tr>
<tr>
<td>20: end</td>
</tr>
<tr>
<td><strong>Output:</strong> LLL-reduced basis (\tilde{b}_1, \tilde{b}_2, \ldots, \tilde{b}_d)</td>
</tr>
</tbody>
</table>

The activation function \(tanh\) is applied on the weighted combination of the input signal \(s_i\), where the weights are entries of the matrix \((H^+_{\text{red}})^\dagger H^+_{\text{red}}\). The activation function generates a soft decision \(v_i^{(l)}\), where \(l = 1\) is the starting iteration, and the maximum number of iterations is \(I_{\text{max}}\). In each iteration the current soft value is subtracted from the soft value from the previous iteration, until the number of iterations exceeds the maximum number of iterations \(I_{\text{max}}\).

### IV. NUMERICAL RESULTS

In this section, we illustrate the behavior of the proposed algorithm in terms of the transmit energy as a function of the ratio \(\alpha = K/N\). Simulation results are compared with the performances of: analytical results for the SE (the number of the redundant representations of each information bit is \(L = 2\)), when \(K \rightarrow \infty\) [13], simulation results for the SE
The number of receive antennas was chosen to be $K$ different channel realizations for the system sizes. The sole HNN algorithm for vector precoding [12]. $K$ different channels. For each realization of the channel matrix the energy penalty exhibited by the HNN. severe performance degradation, the LLL-LR HNN controls the range not model well the RSB problems. We observe that within receive antennas is close to each other, while the HNN does (RSB) [13] in a system where the number of transmit and degrades in comparison to the SE. This is due to the fact and $\alpha$ of the sole HNN algorithm was observed in the range of $\alpha \leq 0.7$. A gradual increase in the performance degradation of the sole HNN algorithm was observed in the range of $0.7 \leq \alpha \leq 0.9$. Within this range the LLL-LR provides performance enhancement, and for $\alpha = 0.9$ the LLL-LR HNN outperforms the sole HNN by 1 dB, 0.9 dB, and 0.4 dB, for $K = 4$, $K = 8$, and $K = 16$, respectively.

For loads close to 1, the performance of the sole HNN degrades in comparison to the SE. This is due to the fact that the lattice precoding exhibits replica symmetry breaking (RSB) [13] in a system where the number of transmit and receive antennas is close to each other, while the HNN does not model well the RSB problems. We observe that within the range $0.9 \leq \alpha \leq 1$, where the sole HNN suffered from severe performance degradation, the LLL-LR HNN controls the energy penalty exhibited by the HNN.

Fig. 2 indicates the performances of the LLL-LR HNN algorithm in comparison to the SE. For example, for $\alpha = 0.9$ and $K = 4$, the SE outperforms the LLL-LR HNN by only 0.2 dB, as indistinguishably well as SE while being only of cubic complexity.

V. COMPUTATIONAL COMPLEXITY

In this subsection we compare the computational complexity of the proposed algorithm, and algorithms used for comparison. The algorithms considered in this work require a matrix inversion to calculate the pseudo-inverse. The calculation of the matrix inverse is of $O(K^3)$ complexity. All the additional complexity introduced in the system, that has complexity less

\begin{algorithm}
\caption{Hopfield Neural Network for Vector Precoding}
\begin{algorithmic}[1]
\State \textbf{Input:} Set $H$, $v = s$, $l = 1$, $I_{\text{max}}$
\State Set $H^+ = H^t (HH^t)^{-1}$
\State Generate LLL $(H^t)$
\State $H_{red} = \text{LLL} (H^t)^U$
\State Set $W = (HH^t)^{-1}$
\EndState
\State $\text{HNN algorithm}$
\State Define: $f_s(y) = -s_j + 2 \tanh(2(y + s_j))$
\State $\textbf{while } l \leq I_{\text{max}} \textbf{ do}$
\State $\textbf{for } 1 \leq j \leq 2K \textbf{ do}$
\State $\text{Calculate}$
\State $v_j^{(i)} = f_s(-\sum_{i=1}^{i-1} w_{ji} v_i^{(i)} - \sum_{i=j+1}^{2K} w_{ji} v_i^{(i-1)})$
\State $\text{end}$
\State $\text{end}$
\State $\textbf{for } 1 \leq j \leq 2K \textbf{ do}$
\State $x_j = -s_j + 2 \cdot \text{sign}(v_j + s_j)$
\State $\text{end}$
\State $\textbf{Output: } x$
\end{algorithmic}
\end{algorithm}

(\alpha = 0.9) and the sole HNN algorithm for vector precoding [12]. The channel matrix $H$ has been modeled with i.i.d. zero-mean Gaussian entries with unit variance. The number of different channel realizations for the system sizes $K = 4$ and $K = 8$ was 10 000, while for $K = 16$ there were 5 000 different channels. For each realization of the channel matrix $H$, the number of the HNN iterations was set to be $I_{\text{max}} = 40$. The number of receive antennas was chosen to be: $K = 4$, $K = 8$, and $K = 16$. Fig. 1 shows the performances of the sole HNN and the LLL-LR HNN precoding algorithms. In Fig. 2 the performances of the LLL-LR HNN precoding algorithm and the SE are compared, for the same simulation setting.

The sole HNN and the LLL-LR HNN algorithms in Fig. 1 provide similar performances for loads in the range $0.5 < \alpha \leq 0.7$. A gradual increase in the performance degradation of the sole HNN algorithm was observed in the range of $0.7 \leq \alpha \leq 0.9$. Within this range the LLL-LR provides performance enhancement, and for $\alpha = 0.9$ the LLL-LR HNN outperforms the sole HNN by 1 dB, 0.9 dB, and 0.4 dB, for $K = 4$, $K = 8$, and $K = 16$, respectively.

For loads close to 1, the performance of the sole HNN degrades in comparison to the SE. This is due to the fact that the lattice precoding exhibits replica symmetry breaking (RSB) [13] in a system where the number of transmit and receive antennas is close to each other, while the HNN does not model well the RSB problems. We observe that within the range $0.9 \leq \alpha \leq 1$, where the sole HNN suffered from severe performance degradation, the LLL-LR HNN controls the energy penalty exhibited by the HNN.

Fig. 2 indicates the performances of the LLL-LR HNN algorithm in comparison to the SE. For example, for $\alpha = 0.9$ and $K = 4$, the SE outperforms the LLL-LR HNN by only 0.2 dB, as indistinguishably well as SE while being only of cubic complexity.

V. COMPUTATIONAL COMPLEXITY

In this subsection we compare the computational complexity of the proposed algorithm, and algorithms used for comparison. The algorithms considered in this work require a matrix inversion to calculate the pseudo-inverse. The calculation of the matrix inverse is of $O(K^3)$ complexity. All the additional complexity introduced in the system, that has complexity less

\begin{table}[h]
\centering
\caption{Hopfield Neural Network for Vector Precoding.}
\begin{tabular}{|c|c|c|c|c|c|}
\hline
\textbf{Algorithm 2: Hopfield Neural Network for Vector Precoding} \\
\hline
1: \textbf{Input:} Set $H$, $v = s$, $l = 1$, $I_{\text{max}}$
2: \textbf{Set} $H^+ = H^t (HH^t)^{-1}$
3: \textbf{Generate} LLL $(H^t)$
4: $H_{red} = \text{LLL} (H^t)^U$
5: \textbf{Set} $W = (HH^t)^{-1}$
\hline
6: \textbf{HNN algorithm}
7: \textbf{Define:} $f_s(y) = -s_j + 2 \tanh(2(y + s_j))$
8: \textbf{while } l \leq I_{\text{max}} \textbf{ do}$
9: \textbf{for } 1 \leq j \leq 2K \textbf{ do}$
10: \textbf{Calculate}$
11: v_j^{(i)} = f_s(-\sum_{i=1}^{i-1} w_{ji} v_i^{(i)} - \sum_{i=j+1}^{2K} w_{ji} v_i^{(i-1)})$
12: \textbf{end}$
13: \textbf{end}$
14: \textbf{for } 1 \leq j \leq 2K \textbf{ do}$
15: x_j = -s_j + 2 \cdot \text{sign}(v_j + s_j)$
16: \textbf{end}$
17: \textbf{Output: } x$
\hline
\end{tabular}
\end{table}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig1.png}
\caption{HNN Algorithm. The average transmit energy as a function of $\alpha$.}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig2.png}
\caption{LLL-LR HNN Algorithm. The average transmit energy as a function of $\alpha$.}
\end{figure}
than cubic, will be negligible in comparison to the matrix inversion.

The LLL algorithm obviously has complexity $O(K^2)$ per iteration. Recent results [17] on the worst and average case complexity of the LLL provided new insights on the complexity of the LLL applied to a wireless MIMO system. It has been shown that the number of the LLL iterations is not upper-bounded, when the lattice reduction is carried out on an i.i.d. Rayleigh channel matrix. The average complexity of the LLL in an i.i.d. Rayleigh channel, in terms of the average number of the LLL iterations, is polynomial with respect to the lattice dimension.

Research on the convergence time of neural networks (e.g., [18], [19], [20] ) has closely followed the active research field on various models and diverse attractive applications of neural networks. There are various realizations of HNN, for example: continuous or discrete time, feedforward or recurrent model, with discrete or analog activation function, finite or infinite network size, asynchronous or synchronous network. The HNN computational complexity has been analyzed depending on the network model and its applications. Some of the results on the computational complexity have been generalized. First we consider the results on the convergence of a symmetric HNN applied for an energy minimization that we used in the proposed algorithm, in terms of convergence time. In the worst case the HNN convergence time may be exponential, but that under some mild conditions, the binary HNN converges in only $O(\log \log K)$ parallel steps in the average case. For the binary HNN with polynomial weight size, convergence time is polynomial with absolute error less then $0.243 \sum_{i=1}^{K} \sum_{j=1}^{K} |w_{ij}|$.

The expected complexity of the SE [7], with respect to the number of jointly detected symbols is exponential. However, the complexity of SE although high, for the applications of the moderate size can be considered as a practical algorithm. CR is based on the application of the convex optimization solvers of polynomial complexity. The CR precoding is carried out by a quadratic solver, that has the computational complexity of $O(K^{3.2})$.

VI. CONCLUSIONS

We have proposed a modification of the HNN for vector precoding, based on LLL, in order to enhance the performance of the HNN precoding algorithm, particularly for loads $0.9 \leq \alpha \leq 1$. The simulation results indicate that the LLL-LR HNN outperforms the sole HNN precoding algorithm for loads within this range, for the simulated number of receive antennas $K$. Generalizations to higher order modulations are trivial if Gray mapping is used. For the load within $0.9 \leq \alpha \leq 1$ the LLL-LR HNN achieves an enhancement in performance between 0.4 dB and 2.6 dB, improving the performances and controlling the energy penalty of the HNN precoding, that are degraded for loads $0.9 \leq \alpha \leq 1$. For $K = 4$ and unit load, LLL-LR HNN performs indistinguishably close to the SE, while being only of cubic complexity.

REFERENCES