A low complexity Soft-Input Soft-Output MIMO detector which combines a Sphere Decoder with a Hopfield Network

Daniel J. Louw, Philip R. Botha, B.T. Maharaj
Department of Electrical, Electronic and Computer Engineering University of Pretoria, Pretoria, South-Africa
Email: djlouw@tuks.co.za, philip.botha@mweb.co.za, sunil.maharaj@up.co.za

Abstract—In this paper, a reduced complexity soft-input soft-output MIMO detector is presented. The detector combines a Sphere Decoder, a Hopfield neural network and an error correction code in an iterative structure (turbo). The simulation results demonstrate that with less computational complexity, the proposed system’s performance equals that of a sphere decoder based max-log-map detector in ideal channel conditions. In correlated channel conditions, the system performs within 0.3 dB of the max-log-map detector at a reduced complexity. An adaptive extrinsic information scaling factor is also introduced to improve performance in non-ideal channel conditions.

Index Terms—Sphere Decoder, Hopfield Network, MIMO, Space-time codes, Space-frequency codes, Space-time-frequency codes

I. INTRODUCTION

Multiple input multiple output (MIMO) is one of the technologies that have been developed in an attempt to address the ever increasing demand for affordable, high speed and reliable wireless communication. MIMO systems offer a significant increase in the capacity of the channel. Together with orthogonal frequency division multiplexing (OFDM), MIMO wireless systems permit the transmission of information over time, space and frequency.

By taking advantage of the available diversity in the channel, the performance of wireless systems in multipath fading can be significantly improved. Codes that are able to extract all the diversity available in a channel, whilst maintaining a transmission rate proportional to the number of transmit antennas have been proposed [1]. These codes make use of Linear Pre-coding (LP) at the transmitter and joint decoding at the receiver. A Sphere Decoder (SD) is generally used to perform the joint decoding of the received encoded symbols.

To improve the performance of the system further, an Error Correction Code (ECC) is generally employed. For best performance, the ECC requires accurate soft outputs from the SD. Various methods have been proposed to calculate the required soft outputs from the SD and they generally fall into two categories: List Sphere Decoding [2] and Max-log-map Hard-to-Soft decoding [3]. In this paper the latter is used.

The Hard-to-Soft decoding makes used of a hard output SD to calculate a maximum a posteriori probability (MAP) solution based on the channel information and the received symbols. For each bit in the received data, another iteration of the hard output SD is run to provide a counter hypothesis to the MAP solution. The soft outputs are then calculated from the MAP solution and the counter hypothesis for each bit.

In this paper the SD that is used to calculate the counter hypothesis is replaced by a suboptimal, though significantly less complex, Hopfield Neural Network (HNN). Due to the suboptimal nature of the HNN, a loss in performance is observed as compared to the pure SD detector. To reduce the loss in performance, the receiver is linked in an iterative turbo structure. The turbo structure allows the SD-HNN based detector to match the performance of the non turbo SD detector at a lower overall complexity.

This paper is structured as follows: In Section II the MIMO detector is described and in Section III the HNN is described. Section IV discusses the Turbo structure and the general system. A complexity analysis is presented in Section V. The simulation parameters and results are given in Section VI. Section VII concludes the paper.

Notation: In this paper we use the following notation. Column vectors (matrices) are denoted by boldface lower (upper) case letters. Superscripts $T$ and $H$ denote the transpose and Hermitian transpose operations, respectively; $\text{diag}(d_1 \ldots d_P)$ denotes a $P \times P$ diagonal matrix with diagonal entries $d_1 \ldots d_P$. $F_P$ is the $P \times P$ discrete Fourier transform (DFT) matrix. $\lfloor x \rfloor$ represents the smallest integer larger than or equal to $x$.

II. MIMO DETECTION

The received signal in a MIMO system may be described as:

$$ y = \mathbf{H}x + n $$

(1)

$$ \Lambda_i^p = LLR(x_i|H,y) $$

$$ \approx \min_{x \in X_i^1} \left\{ \frac{||y + \tilde{y} - \mathbf{H}x||^2}{N_0} \right\} - \min_{x \in X_i^{-1}} \left\{ \frac{||y + \tilde{y} - \mathbf{H}x||^2}{N_0} \right\} $$

(3)

where $\tilde{y}$ is the vector which solves equation 4:

$$ \Lambda^a = 2\mathbf{H}^T \tilde{y}, $$

(4)
where $\mathbf{A}^n$ is the a-priori information on the sequence $\mathbf{x}$ in the LLR format and $X_i^s$ is the set of all possible $\mathbf{x}$ vectors for which $x_i = s$. Solving this equation is done by first calculating the hard ML solution for the entire sequence using a SD. The SD is then run again, once for each bit, to calculate a counter hypothesis. A counter hypothesis is the most likely sequence with the bit in question fixed at the opposite of the ML solution. The ML sequence and the counter hypotheses are then used in (3) to calculate the soft output. This method will be referred to as the Soft-Output SD (SO-SD).

III. HOPFIELD NEURAL NETWORK

HNNs consist of inter-connected neurons. The neurons compute an output $v_i(t)$ from a signal $u_i(t)$ as follows:

$$v_i(t) = g[\beta u_i(t)] \quad (5)$$

where $i$ is the indice of the neuron, $\beta$ is the gain and $g(x)$ is a smooth monotonically decreasing sigmoid function. The signal $u_i(t)$ follows the following equation [4]:

$$\frac{du_i(t)}{dt} = -\frac{u_i(t)}{\tau} + \sum_{j=1}^{N} T_{ij}v_j(t) + I_i \quad (6)$$

where $\tau$ is a constant that can be set equal to unity and $I_i$ is a constant bias added to the input of a neuron. $T_{ij}$ are elements of a symmetric connectivity matrix that are zero when $i = j$ as each neuron connects to all other neurons but not to itself. Hopfield showed that when these conditions are met and the neurons are operating in a high gain mode, large $\beta$, the stable states of the network are the local minima of the computational energy $E$ of the network:

$$E = -\frac{1}{2}v^T \mathbf{T} v - v^T \mathbf{I} \quad (7)$$

Using a HNN to perform ML detection is done by writing the ML equation in a manner as to link the variables in the ML equation with terms in (7) [5], [6].

A. Derivation for the use of a HNN in MIMO detection.

For ML detection, the transmitted vector $\mathbf{x}$ must be found which minimises the detection metric:

$$\mathbf{x} = \arg \min ||\mathbf{y} - \mathbf{H}\mathbf{x}||^2 \quad (8)$$

$$= \arg \min \left\{ \sum_{i=1}^{N} \sum_{j=1}^{N} H_i^T H_j x_i x_j - 2 \sum_{i=1}^{N} x_i H_i^T y \right\} \quad (9)$$

where $H_i$ represents the $i^{th}$ column of $\mathbf{H}$, $x_i$ represents the $i^{th}$ transmitted symbol and $N$ is the number of transmitted symbols. For BPSK type modulation, the values of $\mathbf{x}$ must be either 1 or $-1$. The neuron function must thus meet the requirement:

$$g(-\infty) = -1; \quad g(0) = 0; \quad g(\infty) = 1 \quad (10)$$

One example of such a function is the hyperbolic tan function. This requirement allows one to add the following term to equation 9:

$$-\sum_{i=1}^{N} H_i^T H_i (x_i^2 - 1) \quad (11)$$

Since $x_i$ is always 1 or $-1$ this term will thus always be zero and will not affect the ML equation. We also remove the $i = j$ case from the double summation term in equation 9 and the equation is then simplified as follows:

$$\mathbf{x} = \arg \min \left\{ -2 \sum_{i=1}^{N} x_i H_i^T y + \sum_{i=1}^{N} H_i^T H_i x_i^2 \right\} \quad (12)$$

Simplifying and removing constant terms, the final equation can be given by:

$$\mathbf{x} = \arg \min \left\{ \sum_{i=1}^{N} \sum_{j=0}^{N} H_i^T H_j x_i x_j - 2 \sum_{i=1}^{N} x_i H_i^T y \right\} \quad (13)$$

One can now match the parameters in the Equation (13) to those in the energy equation (7). This is done by defining the following:

$$T_{ij} = \begin{cases} -2H_i^T H_j & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases} \quad (14)$$

$$\mathbf{I} = 2H^T \mathbf{y} \quad (15)$$

This definition results in a symmetric $\mathbf{T}$ matrix where $T_{ij} = T_{ji}$. Substituting these definitions into equation 13 yields the following:

$$\mathbf{x} = \arg \min \left\{ -\frac{1}{2} \mathbf{x}^T \mathbf{T} \mathbf{x} - \mathbf{x}^T \mathbf{I} \right\} \quad (16)$$

which is the energy equation when $\mathbf{x} = \mathbf{v}$. Thus, once the HNN has converged, the transmitted data sequence can be read as the output of the neurons. A discrete time version describing the convergence of the network, equation 6, for use in a digital system is given below:

$$u_i[t+1] = \mathbf{T}_i \mathbf{v}[t] + I_i \quad (17)$$

$$v_i[t+1] = g(\beta u_i[t+1]) \quad (18)$$

where $\mathbf{T}_i$ represents the $i^{th}$ row of $\mathbf{T}$. While this derivation was done for BPSK, the same equations hold for QPSK if a complex to real conversion is performed. Higher order modulation techniques can also be used with the method described in [3].

B. Heuristic Improvements to the HNN

The performance of the HNN may be improved by several heuristic methods. The first considers the update procedure of the network. In (17) the neurons are updated simultaneously. Improved performance can be realised by updating the neurons sequentially where the new outputs are used for updates of the
other neurons. Thus, equation 17 is calculated for $i$ from 1 to $N$ and $v_j[t+1]$ is used instead of $v_j[t]$ if $j < i$.

The second heuristic allows for the inclusion of a momentum term. Normally $\tau = 1$ (see equation 6) which removes the effect of $u[t]$ on $u[t+1]$. Adding a momentum term sets $\tau \neq 1$ and results in $u[t+1]$ depending on $u[t]$. This changes equation 17 to be:

$$u_i[t+1] = \alpha u[t] + T_i v[t] + \gamma I_i$$

(19)

where $\alpha = (1 - \frac{1}{\tau})$ and $\gamma$ are scaling terms which may be varied to improve performance [7]. The third heuristic used to improve performance is called annealing. Annealing has been shown to improve the performance of many kinds of network based optimisation techniques and specifically NNs [8], [9]. In an annealing procedure, variable values are changed from one iteration to the next. The most important parameter to anneal is the gain of the neuron $(\beta)$. Initially, the gain is set low keeping the algorithm from getting stuck in a local minima. The gain is then increased with each iteration to force the algorithm to converge. Annealing may also be used on the other scaling factors introduced above. The values of the parameters and the annealing procedure used in this paper are:

$$\gamma = 0.672 + 0.2979 \exp(0.0575 t - 0.426)$$

(20)

$$\beta = 0.0267 + 0.1363 \exp(0.0442 t - 1.86)$$

(21)

$$\alpha = -0.137 + 0.369 \exp(0.131 t - 1.863)$$

(22)

with $t$ being the iteration number. The parameters were obtained and optimised by using a genetic algorithm to minimise the metric:

$$||y - H\tilde{x}||,$$

(23)

where $\tilde{x}$ is $x$ with bit $\tilde{x}_i = -x_i$.

IV. SYSTEM SPECIFICATIONS

Linear pre-coding of the symbol constellation encodes the data symbols in a manner that extracts the most benefit from the diversity available in the channel. This is done by mapping a set of data symbols to a new set of encoded symbols that are transmitted. This process can be expressed as a matrix multiplication. Let $x = [x_1, \ldots, x_M]^T$ be a data vector of length $M$ complex symbols from a modulation alphabet (eg. QPSK, M-QAM). Let $\Theta$ be a unitary matrix of size $M \times M$. $\Theta$ may be defined as [10]:

$$\Theta = F_M^H \text{diag}(1, \varphi, \ldots, \varphi^{M-1}),$$

(24)

where $\varphi = \exp(j2\pi/4M)$. The linear pre-coding may now be expressed as:

$$s = \Theta x$$

(25)

where $s$ is the resultant encoded vector of complex symbols. A linear pre-coded (LP) data vector $(s)$ permits the correct decoding of all of the data symbols $(x)$ encoded by the linear pre-coder upon the reception of a single encoded symbol $(s_i)$, thus yielding diversity equal to the rank of the rotation matrix $(R_{\Theta})$ [10]. For the remainder of this paper, we will assume the following:

- The channel will be characterised by frequency-selective block-fading conditions, where the channel remains constant for the duration of an OFDM symbol, but changes from one symbol to the next.
- The number of independent paths in the channel $(L)$ is the same for every transmitter receiver pair.
- Perfect Channel State Information (CSI) is available at the receiver and no CSI is available at the transmitter.
- Perfect carrier and symbol synchronisation at the receiver.

The MIMO-OFDM system has $N_T$ transmit and $N_R$ receive antennas. For a given Space Frequency (SF) code, codewords will be placed over $N_F$ frequency tones. Data in binary form is mapped to a symbol in a complex modulation constellation (eg. QPSK). Let $x = [x_1, \ldots, x_M]^T$ be a data vector of length $M$ complex modulation symbols, where $M = N_T N_L$ and $N_L = 2^{[\log_2 L]}$ as in [1]. The data, $x$, then undergoes linear pre-coding as described in equation 25. Placing these encoded symbols appropriately over space and frequency will result in a rate 1 code. However, a rate $N_T$ code may be created by layering $N_T$ such encoded vectors. Layering is achieved by multiplying the $i^{th}$ vector by the diophantine number: $\varphi^{-1}$ [1]. For this paper

$$\phi = \varphi^{1/N_T}$$

(26)

was used where $\varphi = \exp(j2\pi/4M)$. The number of frequency tones required can be given by $N_F = 2^{[\log_2 N_T + \log_2 L]}$. The OFDM time symbols are then created and transmitted. In this paper $N_T$ and $N_R$ were fixed at $N_T = N_R = 2$. The codes were designed for $L = 2$ resulting in $N_F = 4$ and the codes are represented using matrices. The rows correspond to frequency tones, while the columns represent the transmit antennas. The matrix representing the specific code used is given below:

$$\begin{pmatrix}
    s_1(1) & \phi s_2(1) \\
    \phi s_2(2) & s_1(2) \\
    s_1(3) & \phi s_2(3) \\
    \phi s_2(4) & s_1(4)
\end{pmatrix}$$

(27)

where $[s_1(1), \ldots, s_1(4)] = \Theta [x_i(1), \ldots, x_i(4)], i = 1, 2$. $\Theta$ represents the $4 \times 4$ linear pre-coding matrix and $i$ represents the $i^{th}$ layer. This code may also be found in [1]. For this paper the complex symbols were taken from a QPSK modulation alphabet. An SF symbol thus contains 16 bits.

A. Error Correction Code

In principle any ECC could be used to demonstrate the performance of the SD-HNN decoder. In this paper a non-binary quasi-cyclic low density parity check (NB-QC-LDPC) code was chosen. The structure of the parity check matrix of the code was taken from [11]. However, the non-binary elements were chosen randomly from the nonzero elements of the finite field. The code parameters are given in table I. The decoder used is the fast Fourier transform belief propagation (FFT-BP) algorithm [12]–[14].
TABLE I  
NB-QC-LDPC CODE PARAMETERS

<table>
<thead>
<tr>
<th>Code parameters</th>
<th>Code Rate</th>
<th>Code Length (n) in bits</th>
<th>Data Length (k) in bits</th>
<th>Field size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \frac{7}{3} )</td>
<td>2304</td>
<td>1152</td>
<td>256 ( [GF(2^8)] )</td>
</tr>
</tbody>
</table>

B. Turbo System

The performance of the HNN is suboptimal, thus an iterative joint detector and decoder (Turbo) structure was used to improve the performance of the system. A block diagram of the turbo receiver is shown in Figure 1. The turbo process works by passing extrinsic information generated by one decoder to the other decoder to be used as a priori information. In the first iteration the SD decodes the received signal and assumes the a priori information is zero. The SD then passes the MAP solution to the HNN which calculates a counter hypothesis solution for each bit. Using these counter hypotheses in the max-log-map equation given in 3 produces soft outputs which are passed to the FFT-BP decoder. The FFT-BP decoder runs a set amount of decoding iterations. Extrinsic information is calculated from the output of the FFT-BP decoder, which is then passed back to the SD. The SD then re-decodes the received information using the extrinsic information as a-priori information. The resultant soft outputs then represents the a posteriori information. From this information the extrinsic information is calculated and passed to the BP decoder. This iterative process may be repeated as often as required. The extrinsic information \( \Lambda_e \) is calculated as:

\[
\Lambda_e = \Lambda_p - \Lambda_a. \tag{28}
\]

Since the soft MIMO detector is sub-optimal, the extrinsic information coming from the HNN will tend to be overoptimistic (too large). A simple method to counter this and improve the performance is to introduce an extrinsic information scaling factor (EISF). This has been shown to work in many turbo coded systems. An analysis to find a good EISF, denoted by \( \eta \), was done and the results presented in Section VI.

V. COMPLEXITY ANALYSIS

It is difficult to provide an analytical estimate of the complexity of the SD, as it is not constant and heavily influenced by the heuristic methods used in the tree pruning. The results provided in literature places the complexity between \( O(n^6) \) and \( O(n^3) \) in best case scenarios. The complexity of the HNN is however very easy to calculate. The complexity calculation can be broken into two parts: the number of operations required for the initialization process and the number of calculations required per iteration of the network. For this analysis each multiplication or addition will be considered as a floating point operation (flop). The calculation of a transcendental function will be kept separate and referred to as a tflop. These may be calculated using look up tables.

A. Initialization Phase

Setting up the connection weights requires the calculation of equations 14 and 15. This corresponds to one matrix on matrix multiply and one vector on matrix multiply. This complexity is given in Table II.

B. Iteration

Each HNN iteration requires the calculation of equation 19. This requires : one matrix on vector multiply, two scalar on vector multiplies and two vector additions. The annealing of the parameters requires: three transcendental calculations, six multiplies and six additions. An iteration also requires the calculation of the neuron function. This corresponds to one scalar by vector multiplication and one vector transcendental calculation. The number of flops required per iteration will be denoted by \( C_{\text{iter}} \) and the number of tflops required per iteration will be denoted by \( C_{\text{fiter}} \). The complexity is given in Table III.

<table>
<thead>
<tr>
<th>Description</th>
<th>Operations</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix \times Matrix</td>
<td>( n^2 )</td>
<td>flops</td>
</tr>
<tr>
<td>Matrix \times Vector</td>
<td>( 2n^2 )</td>
<td>flops</td>
</tr>
<tr>
<td>Vector + Vector</td>
<td>( 2n )</td>
<td>flops</td>
</tr>
<tr>
<td>Vector Transcendental</td>
<td>( n )</td>
<td>tflops</td>
</tr>
<tr>
<td>Annealing</td>
<td>3</td>
<td>tflops</td>
</tr>
<tr>
<td>Annealing</td>
<td>12</td>
<td>tflops</td>
</tr>
<tr>
<td>( C_{\text{fiter}} )</td>
<td>( 2n^2 + 5n + 12 )</td>
<td>tflops</td>
</tr>
<tr>
<td>( C_{\text{iter}} )</td>
<td>( n + 3 )</td>
<td>tflops</td>
</tr>
</tbody>
</table>

C. Complexity Comparison

For the SO-SD detection, 50 iterations of the FFT-BP algorithm were run. For the SD-HNN detector 10 FFT-BP iterations were run per turbo iteration, resulting in a total of 50 FFT-BP iterations. The complexity of the FFT-BP decoding is thus not considered in this analysis. The total complexity of the SD-HNN detector can be expressed as:

\[
C_{\text{Total}} = i_o(C(SD) + i_i(C_{\text{fiter}} + C_{\text{iter}})) \tag{29}
\]

\[
i_oC(SD) + i_i(C_{\text{fiter}} + C_{\text{iter}}) \tag{30}
\]
where \( i_o \) is the number of turbo iterations and \( i_i \) is the number of internal HNN iterations. \( C(SD) \) represents the complexity of a SD iteration. Notice that the initialisation complexity of the HNN detector has been omitted as the required equations will have been calculated by the SD. The complexity of the SO-SD detector \( (C_{SO-SD}) \) can be represented as:

\[
C_{SO-SD} = (n + 1)C(SD)
\] (31)

The specific complexity gain will thus depend on the number of iterations and \( n \). In this paper we chose \( i_i = 10 \), \( i_o = 5 \). The complexity gain can then be expressed as:

\[
C_{gain} = C_{SO-SD} - C_{Total}
\] (32)

\[
= (n + 1)C(SD) - 5C(SD) - 50(C_{filter} + C_{titer})
\] (33)

\[
= (n - 4)C(SD) - 50(C_{filter} + C_{titer})
\] (34)

Considering that the complexity of the SD is cubic in \( n \) and the complexity of the HNN is square in \( n \), the gain is heavily dependant on \( n \). In this paper \( n = 16 \) and the run-time for the SD-HNN detector was approximately 5 times faster than the SO-SD detector.

VI. Simulation Results

The simulations were performed on the channel simulator developed in [15]. The SD used in the simulations was taken from [16]. The simulation parameters can be found in Table IV. Simulations were performed on an ideal two tap channel and a more realistic suburban alternative channel. The suburban alternative power delay profile (PDP) in Table IV can be found, and was applied to the simulator in [17].

<table>
<thead>
<tr>
<th>TABLE IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIMO-WiMAX Simulation Parameters</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MIMO-OFDM parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmit antennas</td>
<td>2</td>
</tr>
<tr>
<td>Receive antennas</td>
<td>2</td>
</tr>
<tr>
<td>FFT size</td>
<td>128</td>
</tr>
<tr>
<td>Cyclic prefix length</td>
<td>0.25</td>
</tr>
<tr>
<td>Maximum doppler spread</td>
<td>( f_d = 100 \text{Hz} )</td>
</tr>
<tr>
<td>Sampling Time</td>
<td>( T_s = 0.8 \text{µs} )</td>
</tr>
<tr>
<td>Channel Bandwidth</td>
<td>1.25MHz</td>
</tr>
<tr>
<td>Transmit filter</td>
<td>Square root raised cosine, ( \alpha = 0.5 )</td>
</tr>
<tr>
<td>Receive filter</td>
<td>Square root raised cosine, ( \alpha = 0.5 )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ideal Channel Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency-Selectivity</td>
<td>Two-ray equal power PDP at 0 ( \text{µs} ) and 8 ( \text{µs} )</td>
</tr>
<tr>
<td>Time-Selectivity</td>
<td>Slow-fading conditions with ( f_d = 100 \text{Hz} )</td>
</tr>
</tbody>
</table>

A. EISF study

A study was performed to find the best EISF. The result of the study in the uncorrelated two tap channel can be seen in Figure 2 where the \( E_b/N_0 \) required to achieve a BER of \( 10^{-4} \) is plotted against different EISF values. From the figure one can see that an EISF of 0.4 yields the best results. In correlated channel conditions the performance of the system degrades as the capacity of the channel decreases. This means that the \( E_b/N_0 \) range over which the system has to operate is much larger. As a result, the EISF can be further optimised to depend on the \( E_b/N_0 \). A study was done using different EISF values on a 2 tap channel with an exponential antenna correlation factor [18] of 0.6. Figure 3 shows the performance curves. From the figure the best EISF values for different \( E_b/N_0 \) points were found, and an equation interpolated through them. The equation used by the receiver to calculate the EISF is:

\[
\eta = 0.6764 \exp\left(-0.1413 \frac{E_b}{N_0}\right)
\] (35)

where the \( E_b/N_0 \) value is in dB.

B. Two Tap Channel Simulation

Figure 4 shows the performance of the SD-HNN detector on the two tap channel in both correlated and uncorrelated channel conditions. In the uncorrelated case a uniform EISF of 0.4 was used while Equation 35 was used in the correlated case. From the Figure one can see that the SD-HNN detector equals the performance of the SO-SD detector after 5 iterations in the uncorrelated case. In the correlated case, the SD-HNN detector performs approximately 0.3 dB worse than the SO-SD detector. While only one correlation factor was analysed here due to space constraints, similar results can be expected for other correlation factors.

C. Suburban Alternative Channel Simulation

Figure 5 shows the performance of the SD-HNN detector in the 20 tap suburban alternative channel. From the figure,
one can see that the performance of the SD-HNN detector improves with each iteration. After 5 iterations the performance is very close to that of the SO-SD detector although it takes more iterations to exactly equal the performance. In the correlated case, the SD-HNN detector performs approximately 0.5 dB worse than the SO-SD detector.

**Fig. 5.** Performance of SD-HNN detector on the rate-2 SF code with the NB-QC-LDPC code on the suburban alternative channel.

VII. CONCLUSION

In conclusion, a reduced complexity soft output MIMO detector, which combines an SD with an HNN in a turbo structure, was presented. The detector is able to achieve the same performance as a max-log-map optimal detector using only SDs, in ideal channel conditions, at a lower complexity. In channels with many multiple paths and correlation, the performance of the detector degrades slightly compared to the max-log-map optimal detector but still operates at a significantly reduced complexity. An additional advantage of the proposed system is that it allows one to vary the number of iterations which gives the designer the ability to trade of performance versus complexity which can be useful in terms of energy usage.

**REFERENCES**


