# Nonlinear Code Design for Joint Energy and Information Transfer

Mehdi Dabirnia* and Tolga M. Duman*†
*Dept. of Electrical and Electronics Engineering, Bilkent University, 06800 Ankara, Turkey
†School of ECEE, Arizona State University, Tempe, AZ 85287-5706, USA
Email: {mehdi, duman}@ee.bilkent.edu.tr

***Abstract*—Harvesting energy from radio frequency signals along with transmitting data through them is appealing for different wireless communication scenarios such as RFID systems and implantable devices. In this paper, we propose a technique to design nonlinear codes for use in such systems taking into account both energy transmission and error rate requirements. Specifically, we propose using concatenation of a nonlinear trellis code with an outer low density parity check code. Via examples, we observe that our designed codes operate at SNRs $2.4dB$ away from information theoretic limits, and they outperform reference schemes of concatenating LDPC codes with nonlinear memoryless mappers and using classical linear block codes in a time switching mode. We note that it is possible to close the gap to the information theoretic limits further by more sophisticated receiver designs and more complex encoders.**

**Index Terms - RF energy harvesting, joint energy and information transfer, nonlinear codes, low density parity check codes.**

## I. Introduction

Radio Frequency energy harvesting is a wireless power transfer technique which relies on collecting energy from the radiated RF signals at the receiver allowing wireless devices to derive energy from RF signals for use in their information processing and transmission processes. Potential applications of RF energy harvesting can be found in different areas including wireless sensor networks, wireless body networks and wireless charging systems [1].

Wireless energy transfer and wireless information transmission have previously been considered as separate problems. However, recent work on dual use of RF signals for delivering energy and information demonstrates that there is a natural trade-off on the design of such systems [2]. For systems with joint energy and information transfer it is of interest to increase received power levels and information rates at the same time. Using the most energetic symbol all the time is desirable for the first goal, whereas a uniform distribution on the channel input is required to maximize the mutual information over a symmetric noisy channel. Consequently, there is a natural trade-off and the amount of transmitted information and transferred energy cannot be generally maximized at the same time. Varshney in [2] described this fundamental tradeoff between transmission of energy and power through the same signal. He proved that the capacity-energy function is a nonincreasing concave function and obtained closed form expressions for it for several channels such as the noiseless and noisy binary symmetric channel and the Z-channel. He has also shown that for an AWGN channel with a given minimum received energy and maximum input amplitude constraint, the capacity achieving input distribution consists of finite number of mass points [2].

In a related recent study, energy usage of the receiver has been modelled stochastically and then battery overflow and underflow probabilities have been studied using classical codes and constrained run-length limited (RLL) codes [3]. The results show that constrained RLL codes are better suited for the receiver's energy utilization pattern compared to classical unconstrained ones. Binary code design for simultaneous energy and information transfer has been studied in [4] where achievable rates using constrained RLL codes on binary input noisy channels have been investigated. Most of the existing research in the area of joint energy and information transfer is on information theoretic approaches, specifically on capacity energy functions for different channels [2], on performance achievable with RLL codes [3] and achievable rates over some noisy binary channels using RLL codes [4]. Our focus in this paper is design of practical codes for simultaneous energy and information transfer complementing the existing literature.

In order to address the problem of simultaneous energy and information transfer, one approach is to consider transmission of symbols with different energy levels (amplitudes) which can be exploited to satisfy the minimum requirements on the transferred energy. Here, we consider the case of on-off signaling in which only two symbols "0"and "1"are used. Linear codes such as convolutional codes and low density parity check (LDPC) codes have equal density of ones and zeros. Hence, in order to transmit more than $\frac{1}{2}$ (normalized) energy per symbol, there is a need to design nonlinear codes with a desired ones density which provide good error correction capabilities. With this motivation, we propose a coding scheme based on concatenation of a nonlinear trellis code (NLTC) with an outer linear block code, specifically an LDPC code. We describe an algorithm for the inner nonlinear trellis code design. As the outer block code, off-the-shelf LDPC codes optimized for AWGN channels are used, however, it is clear that they too can be optimized for joint energy and information transfer. Via several examples, we observe that our designed codes offer a very good performance, specifically a particular design operates at $2.4dB$ away from the information theoretic limits. The new designs also outperform the reference schemes of concatenating LDPC codes with nonlinear memoryless mappers and using classical linear block codes in a time switching mode. We note that it is possible to close the gap to the information theoretic limits further by more sophisticated (e.g., iterative) receiver designs and by utilizing more complex encoders.

The rest of the paper is organized as follows. In Section II, the channel model is described, information theoretic limits for the considered scenario are given and the proposed scheme of concatenation of an outer linear block code with a nonlinear trellis code is presented. The design of nonlinear trellis codes for our purposes is then introduced in Section III. Ways of avoiding catastrophic codes are discussed in Section IV. In Section V, several numerical examples are provided, and finally, the paper is concluded in Section VI.

## II. Proposed Coding Scheme

### A. Channel Model

We consider an additive white Gaussian noise channel for which the input-output relationship is

$$Y = X + Z, \tag{1}$$

where $X \in \{0,1\}$ and $Z$ is independent and identically distributed (i.i.d.) Gaussian noise with zero mean and variance $\frac{N_0}{2}$. In order to model the trade-off between simultaneous energy and information transfer we need to consider signals with different energy levels. Here, we consider using on-off signaling where "1" (resp. "0") corresponds to the presence (resp. absence) of a signal. Using such a representation enables us to transmit more energy through the channel by using a code with a higher ones density. Signal to noise ratio (SNR) at the receiver side with average ones density $p$ is defined as $SNR = \frac{p}{N_0}$. We assume that the receiver needs to harvest at least a certain amount of energy on average. In order to provide this required energy at the receiver side, we place a constraint on average ones density $p$ at the channel input, i.e., on the coded symbols. Therefore, our aim in this work is to design practical codes with a predetermined constraint on the average ones density of the transmitted codewords.

### B. Information Theoretic Limits

Assuming that the required ones density is $p$ and i.i.d. channel input symbols are used, the capacity of the AWGN channel with a predetermined input distribution (in this case (i.i.d.) Bernoulli($p$) probability mass function) is given by [5]

$$I(X;Y) = h(Y) - \frac{1}{2}log(\pi e N_0), \tag{2}$$

where

$$h(Y) = \int_{-\infty}^{\infty} f_Y(y) log\left(\frac{1}{f_Y(y)}\right) dy, \tag{3}$$

$$f_Y(y) = \frac{1}{\sqrt{2\pi}\sigma}\left((1-p)e^{-\frac{x^2}{N_0}} + pe^{-\frac{(x-1)^2}{N_0}}\right). \tag{4}$$

As an illustration, (2) is computed for $p = \frac{1}{2}$ and $p = \frac{3}{4}$ and the results are shown in Fig. 1 which clearly illustrates that there is a trade-off between the ones density and the maximum possible transmission rate through the channel. That is, by choosing a ones density of $p = \frac{3}{4}$, we can send more energy compared to the uniform input case, however, we sacrifice some data rate.

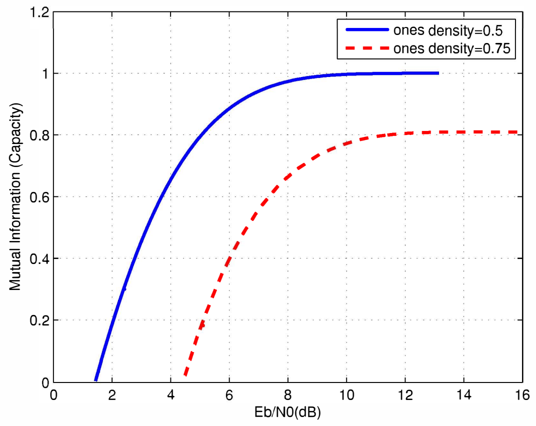

Fig. 1: Capacity of AWGN channel with on-off signaling for $p = \frac{1}{2}$ and $p = \frac{3}{4}$.

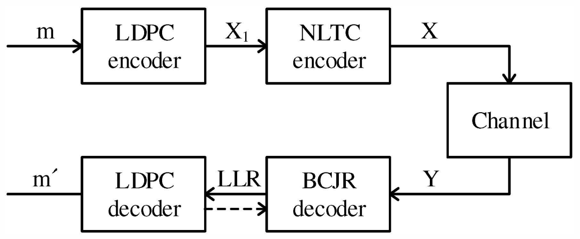

Fig. 2: Block diagram of proposed coding scheme.

### C. Concatenation of LDPC and Nonlinear Trellis Codes

We propose using concatenation of an outer linear block code such as an LDPC code with a nonlinear trellis code as a coding solution for joint energy and information transfer. Fig. 2 shows the block diagram of the proposed coding and decoding scheme. As shown in the figure, message $m$ is first encoded by a rate $R_1$ LDPC code. The output of the LDPC encoder $X_1$ is encoded with a rate $R_2$ (inner) NLTC which results in an overall code rate of $R_1R_2$. At the decoder side, first the noisy output sequence $Y$ is fed into a BCJR decoder corresponding to the NLTC and then the resulting soft information is fed into the LDPC decoder. This soft information is used in a message passing algorithm to obtain estimates of the message bits. Clearly, one can also use turbo equalization and exchange soft information between the BCJR decoder and the LDPC decoder for several iterations to improve the decoding performance as illustrated via the dashed lines in Fig. 2.

In our proposed coding scheme, the design of nonlinear trellis codes is a crucial component, hence we deal with this problem in detail in next two sections.

## III. Nonlinear Trellis Code Design

In this section we present a design technique for nonlinear trellis codes for use over an AWGN channel with the purpose of joint energy and information transfer. Our goal is to maximize the minimum Hamming distance between the codewords through the trellis while keeping a certain ones density. Specifically, we use a sequence of shift registers with $k$ binary inputs at each time instant for a feed-forward convolutional encoder to determine the state transitions through the trellis, and a nonlinear lookup-table to assign encoder outputs for each branch. Encoder outputs are chosen to provide the required ones density $p$. An example of a trellis with memory $M = 3$ and $k = 1$ is shown in Fig. 4. We note that although we do not consider all types of trellises, the specific class that we consider is rich enough to obtain good results for our purposes.

## *A. Generating and Partitioning the Labels*

In this step considering the desired ones density $p$ and code rate $R = \frac{k}{n_\bullet}$ we generate labels with length $n_0$ and average Hamming weight $\omega = pn_0$. Note that to have an average Hamming weight of $\omega$ either we need to select a subset of binary labels with $n_0$ bits that have an average Hamming weight of $\omega$ and then use each of them with the same frequency or to select a subset of labels and use them with different frequencies such that the weighed average Hamming weight is at the desired level. In our approach here, we do not consider the second option, however, we give an example of such a design in Section V. For the rest of the paper we consider code rate of $R = \frac{1}{n_\bullet}$ and simply note that one can carefully generalize these ideas to the case of $R = \frac{k}{n_\bullet}$, $k \neq 1$.

With the goal of maximizing the minimum distance between the codewords, we perform set partitioning on the subset of binary labels selected. In order to do this, we first partition the labels into pairs in such a way that the minimum pairwise Hamming distance between labels in each pair ($d_{min}^{(1)}$) is maximized. Then, we partition the pairs into groups of two pairs such that the pairwise distance between the quadruples ($d_{min}^{(2)}$) is maximized and continue partitioning in this manner. We denote the minimum pairwise Hamming distance of groups of $2^i$ labels with ($d_{min}^{(i)}$). Assuming that the subset of labels has size $2^h$, we obtain a partition tree with $h$ levels. There might be many possibilities to accomplish the partitioning, but at the end we select the one that maximizes $\sum_{i=1}^{h} d_{min}^{(i)}$. An example of a partition tree with $h = 3$ is shown in Fig. 3. Using these labels and applying the extended Ungerboeck's rule (which will be further discussed in Section III-B) we can generate a nonlinear trellis code of memory $M$, where $M + 1 \geq 2h$, with a lower bound on its minimum distance,

$$min - distance \geq 2 \sum_{i=1}^{h} d_{min}^{(i)}. \tag{5}$$

In the case where more than one such subset of labels are available, we select the one that results in the largest lower bound on the minimum distance.

As an example, we consider a design with rate $R = \frac{1}{4}$ and ones density $p = \frac{5}{8}$. Generated labels and a partition tree with three levels plus minimum distance at each level are shown in Fig. 3. For this example, the minimum distance of the code with memory $M \geq 5$ using these labels is $min - distance \geq 10$.

## *B. General Rules for Label Assignment*

The main step in the NLTC design is to assign output values to the branches of the trellis to maximize the minimum distance of the code while keeping the desired ones density. In our

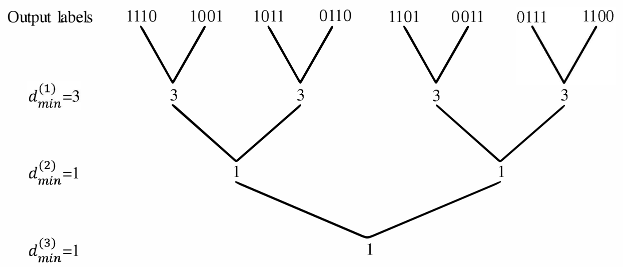

Fig. 3: Generated labels and their partition tree for $R = \frac{1}{4}$ and $p = \frac{5}{8}$ and within distance at different levels.

design, assignment of output labels to branches is performed according to the extended Ungerboeck's rule [6] which tries to maximize the minimum Hamming distance of the code. Ungerboeck noted that every incorrect codeword, in its trellis representation, departs from the correct path (split) and returns to the correct path (merge) at least once, so he maximized the distance between splits and merges. One can extend the Ungerboeck's rule further into the trellis and maximize the Hamming distance between the branches emanating from a split $h$ trellis sections before, where $h$ is a natural number that can be at most $M$. The same procedure can be followed for the branches that merge $h$ sections later. Having the set of $2^h$ distinct labels partitioned in the previous sections for a rate $\frac{1}{n_0}$ code, we can assign the labels according to the extended Ungerboeck's rule as follows:

- $2^i$ labels in the same partition at level $i$ of the partition tree are assigned to $2^i$ branches emanating from (merging at) the same state $i$ trellis sections before (later).
- All the labels are used equally often.

The main difference between our approach for NLTC design and [6] is in the target code rates. That is, the codes designed in [6] are of small rates and are intended for use over a multiple access channel, therefore the sum rate is important for their goal, however, here we design codes of high rates with a specified ones density. One of the design constraints used in [6] is to ensure that all branches produced by the same input to have different output labels which cannot be satisfied for codes of high rates with large memories. Although, we know that in order to design higher rate codes with large minimum distances we should use trellises with large memories. Note that as the memory of the trellis increases, assignment of the labels to branches becomes more complicated, hence we need to have a systematic algorithm to apply the design rules.

## *C. Grouping of Branches for a Specific Trellis*

The aim of this section is to describe an algorithm to arrange the trellis branches in $2^h$ ordered groups (where $h$ is the number of trellis sections for which the Ungerboeck's rule will be extended after each split and before each merge) and to assign partitioned labels to these groups with the same order in which they appear in the partition tree. In the following, we go on with an example to illustrate the grouping step introduced in the previous subsection and then we extend the rules to the

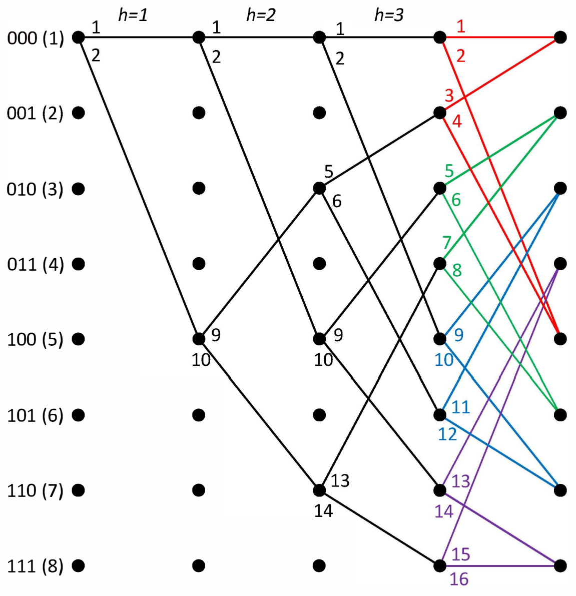

Fig. 4: 8-state trellis diagram and extension of the Ungerboeck's rule.

general case. We consider the 8-state trellis shown in Fig. 4 with $h = 3$ and rate $\frac{1}{n_{\bullet}}$. We number the states in the natural order and assign indexes to each branch as follows: for state number $i \in \{1, 2, ..., 8\}$

$$branch - index = \begin{cases} 2i - 1 & \text{if input } u = 0, \\ 2i & \text{if input } u = 1. \end{cases} \tag{6}$$

First we arrange each group of four branches with consecutive indexes (shown in last section of the trellis in Fig. 4) in two groups and represent them using two blocks $A_l$ and $A_l^*$ as follows

(I) $A_l$

| $a$ | $b$ |
|---|---|
| $4l-3$ | $4l-2$ |
| $4l$ | $4l-1$ |

(II) $A_l^*$

| $a$ | $b$ |
|---|---|
| $4l-2$ | $4l-3$ |
| $4l-1$ | $4l$ |

$\forall l = 1, 2, 3, 4$. The columns of these blocks represent two different groups $a$ and $b$. Here, we define the star operation (*) as exchanging the branches between group $a$ and group $b$. A trivial observation is that $(A^*)^* = A$. Two branches emanating from each split and two branches combining at each merge are placed in different groups inside these blocks, hence the mapping from group indexes to partitioned labels ensures that the Ungerboeck's rule is satisfied for the first section of splits and merges. Using these blocks simplifies grouping of the branches. Considering the second stage, four branches with indexes $\{1, 2, 9, 10\}$ at second section of trellis in Fig. 4 emanating from a split at first section needs to be arranged in different groups, i.e., blocks $A_1$ and $A_3$ should be placed in different groups. Extended rule for the third section after a split forces to arrange eight branches with indexes $\{1, 2, 5, 6, 9, 10, 13, 14\}$ in 8 different groups, i.e., blocks $A_1$, $A_2$, $A_3$ and $A_4$ should be placed in different groups. Note that applying the rule to the third section after splits also satisfies the rule for first and second sections. The same idea can be applied to the merges. Following these rules for this example and using the partitioned labels in Fig. 3 results in the following assignment table.

| Output label | 1110 | 1001 | 1011 | 0110 | 1101 | 0011 | 0111 | 1100 |
|---|---|---|---|---|---|---|---|---|
| Group index | $1a$ | $1b$ | $2a$ | $2b$ | $3a$ | $3b$ | $4a$ | $4b$ |
| Blocks | $A_1$ | | $A_2$ | | $A_3$ | | $A_4$ | |

These rules can be generalized for any $h$, as follows
Split Rule: Place $2^{h-1}$ blocks with indexes $\{i, \frac{S}{2^h} + i, \frac{2S}{2^h} + i, ..., \frac{(2^{h-1}-1)S}{2^h} + i\}$, $\forall i = 1, 2, ..., \frac{S}{2^h}$ in different groups.
Merge Rule: Place $2^{h-1}$ blocks with indexes $\{2^{h-1}i + l | l \in \{1, 2, ..., 2^{h-1}\}\}$, $\forall i = 0, 1, ..., \frac{S}{2^h} - 1$ in different groups.

According to these rules we propose the following algorithm for the grouping step:

1. Decide about $h$ (number of trellis sections for which the Ungerboeck's rule will be extended) and memory of the trellis $M$ (number of blocks will be $2^{M-1}$).
2. Put each of the blocks with indexes 1 to $2^{h-1}$ in groups with same index as the block index, and assign $i = h$.
3. Considering the split rule, put the block with index $2^{i-1} + 1$ in one of the admissible groups (split rule may restrict these admissible placements), then place blocks with indexes $2^{i-1} + 2$ to $2^i$ such that the structure from previous placements of indexes 1 to $2^{i-1} + 1$ is preserved.
4. If there are blocks that are not placed in the groups yet, increment $i$ by one and repeat step 3.

Let us explain the structure mentioned in the step 3 above further. Assume that we have groups with indexes 1 to $2^{h-1}$ at the first level, then at the second level we partition them into groups of two according to their consecutive indexes, i.e., $a_1 = (1, 2)$, $a_2 = (3, 4)$, ..., $a_{2^{h-2}} = (2^{h-1} - 1, 2^{h-1})$, and then at each of the following levels we again partition the groups generated in the previous level into groups of two according to their consecutive indexes and continue this until there is only one group at the final level. An example of this partition of indexes for $h = 3$ is shown in Fig. 5.

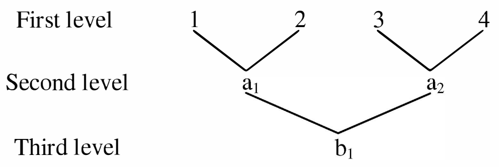

Fig. 5: Partition tree of group indexes for 4 groups ($h = 3$).

Considering this tree one can define the structure as follows: all the changes that occur in the placement of $(2^{i-1}+1)^{th}$ block with respect to the block with index 1, should be applied to all other blocks $2^{i-1}+2$ to $2^i$ with respect to their corresponding block between 2 and $2^{i-1}$. These changes include interchange inside pairs at each level and also the star operation. To make this more clear we give an example in which we have 8 blocks and we want to arrange these blocks in 4 different groups $h=3$. We start by placing blocks with indexes 1 to 4 in groups with the same indexes. Then, at the second step the only freedom that we have is to select between block $A_5$ and $A_5^*$ and place it in one of the admissible groups $\{2,4\}$. For instance, if we select the block $A_5^*$ and place it in group 2, changes that occurred with respect to block $A_1$ by looking at the partition tree (Fig. 5) are the star operation and the interchange inside pairs in the first level. The same set of changes should be applied to all the blocks with indexes 6 to 8 with respect to their corresponding block with indexes 2 to 4. The result is shown in the following table in which group indexes shown in the first row, and the columns represent different groups.

| 1 | 2 | 3 | 4 |
|---|---|---|---|
| $A_1$ | $A_2$ | $A_3$ | $A_4$ |
| $A_6^*$ | $A_5^*$ | $A_8^*$ | $A_7^*$ |

At the end, after completing the grouping of the blocks, we assign distinct labels to each group. We use the partitioned labels obtained previously and assign them to the groups with the same order in which they appear in the partition tree.

## IV. Avoiding Catastrophic Codes

We refer to a code that is prone to catastrophic error propagation as a catastrophic code for which a finite number of channel errors may cause an infinite number of decoder errors. With this reference, it can be seen that a nonlinear trellis code is a catastrophic code if and only if one of these two conditions occur: 1) there is a cycle in its state diagram such that starting from at least two different states of the cycle and traversing around results in the same output sequence but different input sequences, 2) there are at least two different cycles in its state diagram with different input sequence but same output sequences. Such codes clearly need to be avoided to achieve good error correction capabilities, however, checking for this condition when the memory of the trellis grows can be very complicated.

For the specific design (grouping) that we introduced in the previous section, we show that checking for the code being catastrophic can be performed in an easy way by connecting the design to standard convolutional codes. Checking for the catastrophicity of a convolutional code is much simpler, i.e., a convolutional code is catastrophic if its corresponding state diagram contains a circuit in which a nonzero input sequence corresponds to an all-zero output sequence [7]. In the following, we introduce two theorems regarding avoidance of catastrophic codes for our specific design in Section III-C. The proofs are omitted due to space constraints.

*Theorem 1:* For the design in Section III there exists a one-to-one mapping (which is not necessarily unique) from the full set of binary labels with $h$ bits to the group indexes such that assignment of the corresponding binary labels to the branches inside the group results in a linear convolutional code. The resulting convolutional code is called the corresponding convolutional code of the original NLTC.

*Theorem 2:* An NLTC is catastrophic if and only if the corresponding convolutional code is catastrophic.

These two theorems allow us to avoid catastrophic codes in a simple way, and they are utilized in the next section for our specific examples.

## V. Numerical Examples

In this section, we present several examples of the designed nonlinear codes for joint information and energy transfer, and evaluate their performance over an AWGN channel. As a first example, we consider a 16-state trellis $(M=4)$ and a ones density of $p=\frac{3}{4}$, and design NLT codes of rates $\frac{1}{2}$, $\frac{1}{3}$ and $\frac{1}{4}$. For each given rate we need to select a set of binary labels with the corresponding length which satisfies the desired ones density. Selected output labels are shown in Table I. Since there are four labels, we arrange the branches in 4 different groups using the grouping algorithm described earlier. Table II shows the groups of branches and labels assigned to each group (note that the branches are represented by the branch number introduced in (6)). Also the minimum distances of the resulting codes are shown in Table I.

TABLE I: Labels for different rates with $p=\frac{3}{4}$.

| Rate | L#1 | L#2 | L#3 | L#4 | Min-distance |
|---|---|---|---|---|---|
| 1/2 | 01 | 11 | 10 | 11 | 3 |
| 1/3 | 011 | 101 | 110 | 111 | 5 |
| 1/4 | 0111 | 1011 | 1101 | 1110 | 10 |

TABLE II: Example of grouping and label assignment to the baranches of 16-state trellis $(M=4)$ using the proposed algorithm.

| Group index | Branch index | | | | | | | | Output label |
|---|---|---|---|---|---|---|---|---|---|
| 1a | 1 | 4 | 13 | 16 | 22 | 23 | 26 | 27 | L#1 |
| 1b | 2 | 3 | 14 | 15 | 21 | 24 | 25 | 28 | L#2 |
| 2a | 5 | 8 | 9 | 12 | 18 | 19 | 30 | 31 | L#3 |
| 2b | 6 | 7 | 10 | 11 | 17 | 20 | 29 | 32 | L#4 |

We now investigate the performance of the designed codes through our proposed coding scheme. We concatenate an NLTC of rate $R=\frac{1}{2}$, memory $(M=8)$, and ones density $p=\frac{3}{4}$ which has a minimum distance 6 with an outer LDPC code of rate $R=0.93$ and blocklength $32k$. Parity check matrix of the LDPC code used is taken from [8]. The overall rate of this code is $R=0.465$, and its bit error rate performance is illustrated in Fig. 6. Information theoretic results indicate that we need about $6.38dB$ for reliable communication at this transmission rate for this ones density (Fig. 1). Considering that our designed code achieves a bit error rate of $10^{-4}$ at $8.79dB$, we observe a performance about $2.4dB$ away from this limit. We note that some of this gap can be closed by a more sophisticated decoding algorithm, e.g., by utilizing turbo equalization. For comparison purposes, we also consider a reference scheme of using a nonlinear memoryless mapper (NLMM) of the same rate and ones density concatenated with the same LDPC code whose performance is shown in the same figure. We observe that our proposed scheme yields a gain of more than $1dB$ in terms of the BER performance over the reference scheme in this example.

We now compare the performance of the designed codes for joint energy and information transfer with that of classical

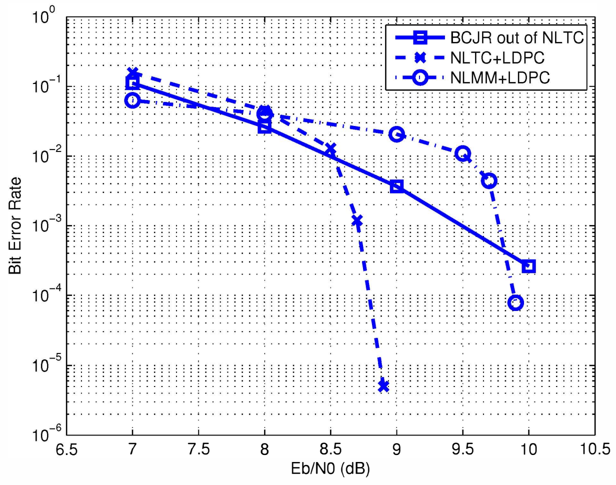

Fig. 6: Bit error rate performance of designed NLTC (memory $(M = 8)$) and nonlinear memoryless mapper both with rate $R = \frac{1}{2}$ and ones density $p = \frac{3}{4}$ and concatenated code with outer LDPC code of rate $R = 0.93$ and block length $32k$.

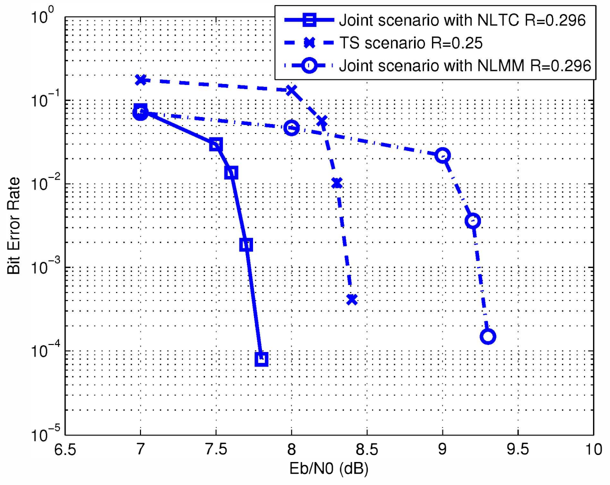

Fig. 7: Comparison of bit error rate performance between joint transfer scenario and time switching scenario

linear codes used with time switching (TS). For the time switching alternative, sending both information and energy half the time using on-off signaling, and sending only "1" for the other half will result in a ones density of $p = \frac{3}{4}$. In this example, degree distributions for LDPC codes are obtained from [9] and parity check matrices are obtained with tools in [10]. An irregular LDPC code of rate $\frac{1}{2}$ with blocklength $32k$ is used for the TS option, hence the overall information transfer rate is 0.25. As an example of our proposed design, a nonlinear code realized with concatenation of an NLTC of rate $\frac{1}{3}$, memory $m = 8$ and ones density $p = \frac{3}{4}$ with an outer LDPC code of rate $\frac{8}{9}$ and blocklength $32k$ (resulting in an overall rate of 0.296 which is higher) is used for joint energy and information transfer. Also considered is the reference scheme of using NLMM instead of NLTC. Fig. 7 shows the bit error rate performance results of these three scenarios. It can be observed that using the designed nonlinear codes for joint energy and information transfer provides clear rate and SNR benefits compared to linear codes in a time switching mode and to the scheme using a memoryless mapper.

## VI. Conclusion

In this paper, a coding scheme based on concatenation of a nonlinear trellis code with an outer LDPC code for joint energy and information transfer is proposed. An algorithm is developed to design the inner NLTC, and ways of checking whether the code is catastrophic or not are presented. Several examples of the proposed design are provided, and their performances are evaluated. Simulation results show that designed codes beat the alternative of using classical linear block codes with time switching and the reference schemes of concatenating LDPC codes with nonlinear memoryless mappers, however, there is still some gap to the information theoretic limits which can be reduced by more sophisticated receiver designs and by the use of more complex encoders.

## Acknowledgment

This publication was made possible by NPRP Grant 4-1293-2-513 from Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

## References

[1] X. Lu, P. Wang, D. Niyato, D. I. Kim, and Z. Han, "Wireless networks with RF energy harvesting: A contemporary survey," *arXiv:1406.6470,* Jun. 2014. [online]. Available: http://arxiv.org/abs/1406.6470)

[2] L. R. Varshney, "Transporting information and energy simultaneously," in *Proc. of IEEE Int. Symp. on Inform. Theory,* pp. 1612–1616, Jul. 2008.

[3] A. M. Fouladgar, O. Simeone, E. Erkip, "Constrained Codes for Joint Energy and Information Transfer," *IEEE Trans. Commun.,* vol. 62, no. 6, pp. 2121–2131, Jun. 2014.

[4] A. Tandon, M. Motani, and L. Varshney, "On code design for simultaneous energy and information transfer," in *Inform. Theory and Applicat. Workshop (ITA)* pp. 1–6, Feb. 2014.

[5] T. M. Cover and J. A. Thomas, *Elements of Information Theory,* 2nd Edition. New York, USA: Wiley, 2006.

[6] M. Griot, A. I. V. Casado, W.-Y. Weng, H. Chan, J. Wang, and R. D. Wesel, "Nonlinear trellis codes for binary-input binary-output multiple access channels with single-user decoding," *IEEE Trans. Commun.*, vol. 60, no. 2, pp. 364-374, Feb. 2012.

[7] S. B. Wicker, *Error Control Systems for Digital Communication and Storage,* Englewood Cliffs: Prentice Hall, 1995.

[8] D. J. C. MacKay, "Encyclopedia of Sparse Graph Codes," 1999, [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/Codes-Files.html

[9] T. J. Richardson, A. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory,* vol. 47, no. 2, pp. 619–637, Feb. 2001.

[10] [Online]. Available: http://itpp.sourceforge.net/4.3.1/