
CHALMERS
UNIVERSITY OF TECHNOLOGY

# Decoding Reed-Muller Codes Using Minimum- Weight Parity Checks

Downloaded from: https://research.chalmers.se, 2024-10-13 00:45 UTC

Citation for the original published paper (version of record):
Santi, E., Häger, C., Pfister, H. (2018). Decoding Reed-Muller Codes Using Minimum- Weight Parity Checks. IEEE International Symposium on Information Theory - Proceedings, 2018-June: 1296-1300. http://dx.doi.org/10.1109/ISIT.2018.8437637

N.B. When citing this work, cite the original published paper.

© 2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, or reuse of any copyrighted component of this work in other works.

(article starts on next page)

# Decoding Reed–Muller Codes Using Minimum-Weight Parity Checks

Elia Santi*, Christian Häger†‡, and Henry D. Pfister‡
*Department of Information Engineering, University of Parma, Parma, Italy
†Department of Electrical Engineering, Chalmers University of Technology, Gothenburg, Sweden
‡Department of Electrical and Computer Engineering, Duke University, Durham, North Carolina

***Abstract*—Reed–Muller (RM) codes exhibit good performance under maximum-likelihood (ML) decoding due to their highly-symmetric structure. In this paper, we explore the question of whether the code symmetry of RM codes can also be exploited to achieve near-ML performance in practice. The main idea is to apply iterative decoding to a highly-redundant parity-check (PC) matrix that contains only the minimum-weight dual codewords as rows. As examples, we consider the peeling decoder for the binary erasure channel, linear-programming and belief propagation (BP) decoding for the binary-input additive white Gaussian noise channel, and bit-flipping and BP decoding for the binary symmetric channel. For short block lengths, it is shown that near-ML performance can indeed be achieved in many cases. We also propose a method to tailor the PC matrix to the received observation by selecting only a small fraction of useful minimum-weight PCs before decoding begins. This allows one to both improve performance and significantly reduce complexity compared to using the full set of minimum-weight PCs.**

## I. Introduction

Recently, the 5G cellular standardization process focused on error-correcting codes and decoders that are nearly optimal for short block lengths (e.g., rate-$1/2$ binary codes with lengths from $128$ to $512$). Promising contenders include modified polar codes, low-density parity-check (LDPC) codes, and tail-biting convolutional codes [1]–[3]. These results also show that short algebraic codes such as Reed–Muller (RM) and extended BCH (eBCH) codes under ML decoding tie or outperform all the other choices. ML performance can be approached with methods based on ordered-statistics decoding [4] such as most-reliable basis (MRB) decoding [5]. Depending on the code and decoder parameters, the complexity of these methods range from relatively practical to extremely complex.

Motivated by the good performance of RM codes under ML decoding and the proof that RM codes achieve capacity on the binary erasure channel (BEC) [6], we revisit the question of whether code symmetry can be used to achieve near-ML performance in practice. This question, in various forms, has been addressed by several groups over the past years [7]–[9]. In general, one applies a variant of belief-propagation (BP) decoding to the Tanner graph defined by a redundant parity-check (PC) matrix for the code. Often, the redundancy in the PC matrix is derived from the symmetry (i.e., automorphism group) of the code. Methods based on redundant PC matrices are also related to earlier approaches that adapt the PC matrix during decoding [10]–[12].

Highly-symmetric codes such as RM codes can have a very large number of minimum-weight (MW) PCs. The main contribution of this paper is to show that this large set of MWPCs can be exploited to provide near-ML performance with several well-known iterative decoding schemes. In particular, we consider the peeling decoder (PD) for the BEC, linear-programming (LP) and BP decoding for the binary-input additive white Gaussian noise (AWGN) channel, and bit-flipping (BF) and BP decoding for the binary symmetric channel (BSC). It is worth noting that the idea of using all MWPCs for decoding appears in [13], even before the rediscovery of iterative decoding. Their decoding algorithms are variations of the BF algorithm [14] applied to a redundant PC matrix. As we will see, the BF algorithms they propose work remarkably well for the BSC, but result in a performance loss compared to LP and BP decoding for the AWGN channel.

Using all available MWPCs for decoding can become quite complex, e.g., the rate-1/2 RM code of length 128 has 94,488 MWPCs. To address this problem, we propose a method to select only a small fraction of useful MWPC based on the channel reliabilities. We exploit the fact that, given any small set of bit positions, there exist efficient methods to find a MWPC that contains the chosen bits. This process is iterated to design a redundant PC matrix that is tailored to the received observation from the channel. The resulting PC matrix can allow one to both improve performance (by reducing cycles in the Tanner graph) and reduce complexity. We stress that this approach works by adapting the PC matrix *before* decoding begins. Thus, it is closer in spirit to MRB decoding than to the adaptive methods employed by [10]–[12].

## II. Background on Reed–Muller Codes

RM codes were introduced by Muller in [15]. We use $\mathrm{RM}(r,m)$ to denote the $r$-th order RM code of length $n=2^m$, where $0 \le r \le m$. Each codeword in $\mathrm{RM}(r,m)$ is defined by evaluating a multivariate polynomial $f \in \mathbb{F}_2[x_1,\ldots,x_m]$ of degree at most $r$ at all points in $\mathbb{F}_2^m$ [16]. The code

This work was done while E. Santi was visiting Duke University, Durham, North Carolina. The work of C. Häger was supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant No. 749798. The work of H. D. Pfister was supported in part by the NSF under Grant No. 1718494. Any opinions, findings, conclusions, and recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors. Corresponding author e-mail: henry.pfister@duke.edu.

$\mathrm{RM}(r,m)$ has minimum distance $d_{\min} = 2^{m-r}$ and dimension $k = \binom{m}{0} + \cdots + \binom{m}{r}$.

### A. Number of Minimum-Weight Parity Checks

For a binary linear code, the codewords of the dual code define all valid rows of the PC matrix. Since the dual code of $\mathrm{RM}(r,m)$ is $\mathrm{RM}(m-r-1,m)$, the MWPCs of $\mathrm{RM}(r,m)$ thus have weight $2^{m-(m-r-1)} = 2^{r+1}$. In order to determine the number of MWPCs for $\mathrm{RM}(r,m)$, one may use the fact that each MW codeword of $\mathrm{RM}(m-r-1,m)$ is the indicator vector of an $(r+1)$-dimensional affine subspace of $\mathbb{F}_2^m$ [16]. Based on this, one can show that the number of MWPCs is given by [16]

$$F(r,m) = 2^{m-r-1} \prod_{i=0}^{r} \frac{2^{m-i}-1}{2^{r+1-i}-1}. \tag{1}$$

For example, the $[128, 64, 16]$ code[1] $\mathrm{RM}(3,7)$ has 94,488 weight-16 PCs.

### B. Generating Minimum-Weight Parity Checks

The connection between MWPCs and affine subspaces also provides an efficient method for generating a MWPC that is connected to any given set of $r+2$ codeword bits. In particular, one can simply complete the affine subspace containing the chosen $r+2$ points. If the chosen set of points is not affinely independent, then one can extend the set to define an $(r+1)$-dimensional affine subspace. This procedure is described in Algorithm 1. The algorithm will be used to construct a PC matrix for $\mathrm{RM}(r,m)$ that is tailored to a particular received sequence. This procedure is described in the next section.

## III. Parity-Check Matrix Adaptation

The PC matrix containing all MW dual codewords as rows is denoted by $\boldsymbol{H}_{\text{full}}$. This matrix can be used directly for iterative decoding, e.g., BP decoding. In general, however, the decoding complexity for the considered iterative schemes scales linearly with the number of rows in the PC matrix. Thus, depending on the RM code, decoding based on the full matrix $\boldsymbol{H}_{\text{full}}$ may result in high complexity.

On the other hand, not all the rows of $\boldsymbol{H}_{\text{full}}$ are equally useful in the decoding of a particular received observation vector $\boldsymbol{y} = (y_1, \ldots, y_n)^\top$. For instance, if all the bits involved in a given PC are relatively unaffected by the channel, the associated PC would be uninformative for the decoding process. Therefore, our approach to reduce complexity is to pick only the rows of $\boldsymbol{H}_{\text{full}}$ that are expected to be useful for the decoding. The choice of rows is based on $\boldsymbol{y}$ and the resulting PC matrix containing the subset of rows is denoted by $\boldsymbol{H}_{\text{sub}}$.

[1]A linear code is called an $[n,k,d]$ code if it has length $n$, dimension $k$, and minimum-distance $d$.

**Algorithm 1** For $\mathrm{RM}(r,m)$, generate MWPC $\boldsymbol{w} \in \mathbb{F}_2^n$ with ones in bit positions $i_1, \ldots, i_{r+2} \in \{1, \ldots, n\}$, where $n = 2^m$

Let row-vector $\boldsymbol{v}_j \in \mathbb{F}_2^m$ be the binary expansion of $i_j - 1$
Form matrix $\boldsymbol{A} \in \mathbb{F}_2^{(r+1)\times m}$ with rows $\boldsymbol{a}_j = \boldsymbol{v}_j \oplus \boldsymbol{v}_{r+2}$
$\boldsymbol{B} \leftarrow$ reduced row echelon form of $\boldsymbol{A}$
**while** $\boldsymbol{B}$ contains all-zero rows **do**
    In first column that is not equal to a unit vector, add a one at row position of the first all-zero row
**end while**
Initialize $\boldsymbol{w} \in \mathbb{F}_2^n$ to the all-zero vector
**for all** $l \in \{0, \ldots, 2^{r+1}-1\}$ **do**
    $\boldsymbol{u}_l \leftarrow m$-bit binary expansion of $l$
    $\boldsymbol{z}_l \leftarrow \boldsymbol{u}_l \boldsymbol{B} \oplus \boldsymbol{v}_{r+2}$
    $s_l \leftarrow$ integer represented by binary expansion $\boldsymbol{z}_l$
    $\boldsymbol{w}_{s_l+1} \leftarrow 1$
**end for**

### A. General Idea

In order to illustrate the general idea, suppose the codeword bits are transmitted through a channel and the received values are classified either as *good* or *bad*. For example, on the BEC an unerased bit would be labeled good while an erased bit would be called bad. Then, Algorithm 1 can be used to generate a MWPC that contains one bad bit of interest, $r+1$ randomly chosen good bits, and some set of $2^{r+1}-r-2$ other bits. From an information-theoretic point of view, this MWPC is expected to provide more information about the bad bit of interest than a random MWPC because it involves a guaranteed number of $r+1$ good bits. Repeating this process allows one to generate a PC matrix for $\mathrm{RM}(r,m)$ that is biased towards informative MWPCs.

### B. Reliability Criterion

As a first step, the bit positions $I = \{1, 2, 3, \ldots, n\}$ are divided into two disjoint sets $G$ and $B$ based on their reliability. To that end, one first computes the vector of log-likelihood ratios (LLRs) $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_n)^\top \in \mathbb{R}^n$ based on the received vector $\boldsymbol{y}$. The vector $\boldsymbol{\gamma}$ is then sorted, i.e., $(t_1, \ldots, t_n)$ is a permutation of bit indexes such that $i > j \Rightarrow |\gamma_{t_i}| \geq |\gamma_{t_j}|$. We then set $G = \{t_k \in I : k \leq fn\}$ and $B = I - G$, where $0 \leq f \leq 1$ is a tunable parameter and $fn$ is assumed to be an integer.

*Remark* 1. The LLR sorting can be applied for an arbitrary binary-input memoryless channel with the exception of the BSC. The BSC is discussed separately in Sec. V-C below.

*Remark* 2. The use of sorting may be avoided by instead thresholding the LLRs. However, our numerical studies showed that this results in some loss for the short block-lengths we considered.

### C. Tailoring $\boldsymbol{H}_{sub}$ to the Received Vector

The sets of reliable and unreliable bit positions $G$ and $B$ are then used to generate an overcomplete PC matrix that is tailored to the received vector $\boldsymbol{y}$. The proposed method is

**Algorithm 2** For index sets $G/B$ of good/bad bits, generate a tailored PC matrix $\boldsymbol{H}_{\text{sub}}$ with $s$ rows

```
1: Initialize H_sub to an empty matrix
2: while H_sub has less than s rows do
3:     for all b ∈ B do
4:         Draw {g_1, ..., g_{r+1}} random positions from G
5:         Generate MWPC w based on {b, g_1, ..., g_{r+1}}
6:         if w is not already a row in H_sub then
7:             Append row w to H_sub
8:         end if
9:     end for
10: end while
```

illustrated in Algorithm 2 below, where $s \in \mathbb{N}$ is the targeted number of rows of $\boldsymbol{H}_{\text{sub}}$. Essentially, one iterates through the set of unreliable bit positions $B$, pairing at each step one unreliable bit with $r+1$ reliable ones. Based on the resulting set of $r+2$ bit positions, Algorithm 1 is then used to generate a MWPC (line 5). The generated MWPC is accepted if it does not already exist in $\boldsymbol{H}_{\text{sub}}$. We remark that the if-condition in line 6 of Algorithm 2 can be implemented very efficiently by applying a hashing function to the vector $\boldsymbol{w}$ and storing the result in a hashtable.

## IV. Decoding Algorithms

In this section, we briefly review the decoding algorithms that are used in this paper.

### A. Peeling Decoder

The PD is an iterative decoder for binary linear codes transmitted over the BEC [14]. It operates on the PC matrix of the code and tracks whether the value of each bit is currently known or unknown. If there is a PC equation with exactly one unknown bit, then the value of that bit can be computed from the equation and the process is repeated. Once there is no such PC, the algorithm terminates.

### B. Belief Propagation

BP decoding is an iterative method for decoding binary linear codes transmitted over memoryless channels [14]. It works by passing messages along the edges of the Tanner graph. If the graph is a tree, then BP decoding produces optimal decisions. In general, it is suboptimal and its loss in performance is often attributed to cycles in the graph.

For a code whose Tanner graph has many cycles, it is known that introducing a scaling parameter can improve performance [7]. When using a redundant PC matrix, this can also be motivated by the existence of correlations between input messages to a bit node. Since BP is based on an independence assumption, these correlations typically cause it to generate bit estimates that are overconfident. If these messages are represented by LLRs, then this overconfidence can be reduced by scaling messages by a constant less than one. This approach was also proposed to mitigate the overconfidence associated with min-sum decoding [17]. In this work, the input messages to the bit nodes are scaled by the factor $w$.

### C. Linear-Programming Decoding

LP decoding was introduced in [18]. It is based on relaxing the ML decoding problem into the linear program

$$\min \sum_{i=1}^{n} x_i \gamma_i \quad \text{subject to} \quad \boldsymbol{x} \in \bigcap_{j \in \mathcal{J}} \mathcal{P}_j,$$

where $\mathcal{P}_j$ denotes the convex hull of all $\{0,1\}^n$ vectors that satisfy the $j$-th PC equation. If the solution vector lies in $\{0,1\}^n$, then it is the ML codeword. In theory, a nice property of LP decoding is that the answer is static and does not depend on the presence of cycles in the Tanner graph. But, in practice, solving the LP with conventional solvers can be slow and cycles may affect the convergence speed.

We employ LP decoding using the alternating direction method of multipliers (ADMM), as proposed in [19]. The method is based on an augmented Lagrangian which is parameterized by a tunable scaling parameter $\mu > 0$ [19, Eq. (3.2)]. Then, LP decoding can be implemented as a message-passing algorithm with an update schedule similar to BP, where the update rules can be found in [19, Sec. 3.1]. The algorithm stops after $T_{\text{max}}$ iterations or when a valid codeword is found.

### D. Bit-Flipping Decoding

BF is an iterative decoding method for binary linear codes transmitted over the BSC [14]. In its simplest form, it is based on flipping a codeword bit that maximally reduces the number of PC equations that are currently violated. In one case, we also compare with the weighted BF (WBF) decoder proposed in [13]. This extends the idea to general channels by including weights and thresholds to decide which bits to flip.

### E. Most-Reliable Basis Decoding

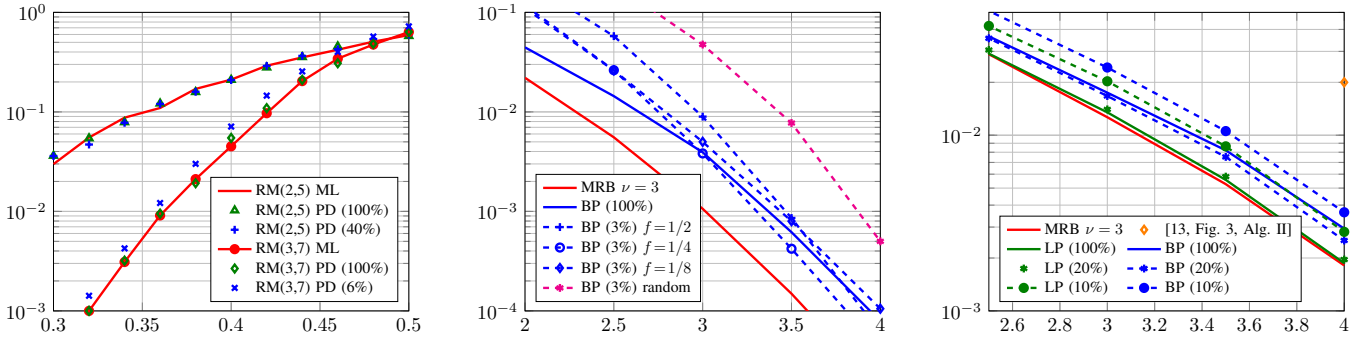
MRB decoding, which was introduced by Dorsch in 1974 [5], is based on sorting the received vector according to reliability (similar to Algorithm 2). After sorting, it uses linear algebra to find an information set (i.e., a set of positions whose values determine the entire codeword) containing the most reliable bits. Then, it assumes there are at most $\nu$ errors in the $k$ reliable positions. Then, one can encode the information set implied by each of these error patterns and generate a list of $\binom{k}{\nu}$ candidate codewords. Finally, the ML decoding rule is used to choose the most-likely candidate codeword.

## V. Numerical Results

For the numerical results, we consider various RM codes with length $n = 32$ and $n = 128$. The code parameters are summarized in Table I. For all data points, at least 100 codeword errors were recorded.

### A. The Binary Erasure Channel

For a linear code on the BEC, the complexity of ML decoding is at most cubic in the block-length [14]. Still, the BEC provides a useful proving ground for general iterative-decoding schemes. In this section, we evaluate the PD for RM codes with redundant PC matrices derived from the complete set of MWPCs.




(a) Block-failure rate versus channel-erasure rate on the BEC for $\mathrm{RM}(2,5)$ and $\mathrm{RM}(3,7)$

(b) Block-error rate versus $E_b/N_0$ in AWGN for $\mathrm{RM}(3,7)$ with $w = 0.05$

(c) Block-error rate versus $E_b/N_0$ in AWGN for $\mathrm{RM}(2,5)$ with $w = 0.2$ and $f = 1/4$

Fig. 1. Results of three numerical performance comparisons

For brevity, we focus only on the rate-1/2 codes $\mathrm{RM}(2,5)$ and $\mathrm{RM}(3,7)$. Fig. 1a shows simulation results for ML decoding, the PD with all MWPCs (100%), and the PD when the PC matrix is tailored to the received sequence using a fixed fraction of available MWPCs. Note that for the BEC, the sets $G$ and $B$ used in Algorithm 2 are simply the unerased and erased bits, i.e., the LLR sorting is not employed.

For the shorter code, all three curves are quite close together. For the longer code, the PD matches the ML decoder when the full set of MWPCs is used. The performance loss of the low-complexity decoder is relatively small for the range tested.

### *B. The Binary-Input Additive White Gaussian Noise Channel*

For the binary-input AWGN channel, we consider both LP and BP decoding. For the LP decoding, the ADMM solver is employed with parameters $\mu = 0.03$ and $T_{\max} = 1000$. For BP, we perform $\ell = 30$ iterations and the weighting factor $w$ is optimized for each scenario. As a comparison, we use MRB with $\nu = 3$ to approximate ML performance.

First, we fix the number of rows in $\boldsymbol{H}_{\text{sub}}$ and illustrate how different strategies for picking MWPCs affect the performance under BP decoding. To that end, the code RM(3,7) is considered with $s = 2835$, i.e., $\boldsymbol{H}_{\text{sub}}$ contains $s/F(3,7) \approx 3\%$ of the complete set of MWPCs. Simulation results are presented in Fig. 1b. The performance is shown for three different values for the parameter $f$. The proposed tailoring strategy leads to better performance compared to picking a random set of MWPCs, with a performance gain up to around 0.5 dB at a block-error rate of $10^{-3}$. It can also be seen that for an optimized choice of $f = 1/4$, the tailoring strategy leads to a better performance compared to using the full set of MWPCs. This can be attributed to the reduced number of cycles in the Tanner graph for $\boldsymbol{H}_{\text{sub}}$ compared to $\boldsymbol{H}_{\text{full}}$. In the following, we fix $f = 1/4$ for all other simulations, noting that it may be possible to increase performance by re-optimizing $f$ for each considered case. We also remark that for LP decoding, similar observations regarding the optimal value of $f$ can be made and the results are omitted.

TABLE I
CODE PARAMETERS

| code | $n$ | $k$ | $d_{\min}$ | $d^{\perp}_{\min}$ | rate | $F(r,m)$ |
|---|---|---|---|---|---|---|
| $\mathrm{RM}(2,5)$ | 32 | 16 | 8 | 8 | 0.5 | 620 |
| $\mathrm{RM}(2,7)$ | 128 | 29 | 32 | 8 | 0.23 | 188,976 |
| $\mathrm{RM}(3,7)$ | 128 | 64 | 16 | 16 | 0.5 | 94,488 |
| $\mathrm{RM}(4,7)$ | 128 | 99 | 8 | 32 | 0.77 | 10,668 |

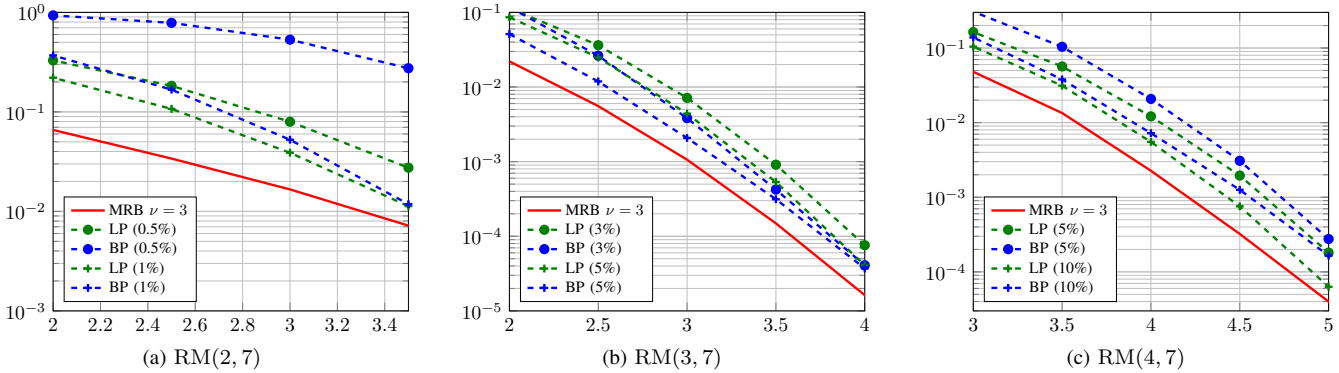
Simulation results for $\mathrm{RM}(2,5)$ using LP and BP decoding are shown in Fig. 1c. For this code, LP decoding outperforms BP decoding and gives virtually identical block-error rates as MRB decoding using both $\boldsymbol{H}_{\text{full}}$ and $\boldsymbol{H}_{\text{sub}}$ with 20% of available MWPCs. For this case, it can be seen again that BP decoding benefits from using a sub-sampled PC matrix $\boldsymbol{H}_{\text{sub}}$ compared to using $\boldsymbol{H}_{\text{full}}$. Also, a simulation point based on WBF is included from [13] to show the superiority of LP and BP decoding. Comparing with [8, Fig. 5], these curves nearly match the ML results for the [31,16,7] BCH code and our "BP (10%)" result is quite close to their "MBBP $l = 6$" curve.

Lastly, we study the performance of three RM codes with length $n = 128$ and a range of rates. The performance is shown in Fig. 2 for a varying number of rows in $\boldsymbol{H}_{\text{sub}}$. From these graphs, we see that the best performance is achieved at a different fraction of rows for each code. In particular, one requires around 1890 rows for $\mathrm{RM}(2,7)$ (1% of $\boldsymbol{H}_{\text{full}}$), 4724 rows for $\mathrm{RM}(3,7)$ (5% of $\boldsymbol{H}_{\text{full}}$), and 1067 rows for $\mathrm{RM}(4,7)$ (10% of $\boldsymbol{H}_{\text{full}}$). These values were chosen so that the performance of the best scheme was roughly 0.25 dB from MRB at a block-error rate of $10^{-2}$.

### *C. The Binary Symmetric Channel*

While the BEC reveals the locations where bits are lost and the binary-input AWGN channel gives soft information for each bit, the BSC provides no indication of reliability for received bits. As we saw in the last section, the performance in AWGN actually improves when a subset of more reliable MWPCs are used for BP decoding. On the BSC, however, it is not possible to advantageously select PCs for BP decoding. Similarly, decoders based on ordered statistics provide no gains on the BSC. Therefore, MRB cannot be used to provide a reference curve for the approximate ML performance.

We consider both BP and BF decoding for the BSC using $\boldsymbol{H}_{\text{full}}$, i.e., the full set of MWPCs. Fig. 3 shows our simulation




(a) $\mathrm{RM}(2, 7)$ (b) $\mathrm{RM}(3, 7)$ (c) $\mathrm{RM}(4, 7)$

Fig. 2. Block-error rate versus $E_b/N_0$ in AWGN with $w = 0.05$ and $f = 1/4$.

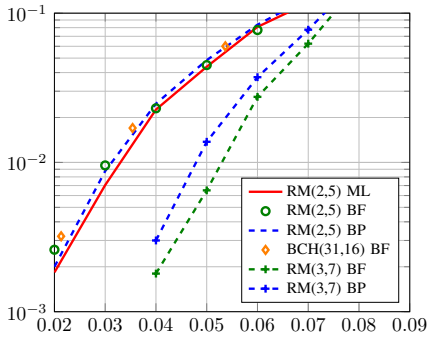



Fig. 3. Block-error rate versus channel-error rate on the BSC for $w = 0.08$

results for $\mathrm{RM}(2, 5)$ and $\mathrm{RM}(3, 7)$. The BF decoder is identical to Algorithm II in [13]. For $\mathrm{RM}(2, 5)$, both the BP and BF decoder perform very close to the ML decoder. The figure also includes results from [13] for the $[31, 16, 7]$ BCH code under BF decoding. One can see that the results for the $[31, 16, 7]$ BCH and the $[32, 16, 8]$ RM code are very similar. This is not surprising because the two codes are nearly identical, i.e., the $[32, 16, 8]$ eBCH code is equivalent to the code $\mathrm{RM}(2, 5)$. Interestingly, for the longer code $\mathrm{RM}(3, 7)$, the BF decoder outperforms the BP decoder with an optimized weight factor.

## VI. Conclusions

We have investigated the iterative decoding of RM codes based on redundant PC matrices whose rows contain MWPCs. Various iterative schemes were considered for the BEC, binary-input AWGN channel, and the BSC. For the $[32, 16, 8]$ code $\mathrm{RM}(2, 5)$, near-ML performance can be achieved using the PD on the BEC, LP decoding on the AWGN channel, and BF or BP decoding on the BSC. For RM codes with $n = 128$ on the BEC, the PD remained very close to optimal. For the AWGN channel, the performance gap of LP and BP decoding with respect to ML decoding increases. It was also shown that, for all channels with the exception of the BSC, the complexity can be reduced by using only a fraction of the available MWPCs. For BP, this strategy also translates into better performance by reducing cycles.

## References


[1] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inform. Theory*, vol. 61, no. 5, pp. 2213–2226, 2015.

[2] G. Liva, L. Gaudio, T. Ninacs, and T. Jerkovits, "Code design for short blocks: A survey," *arXiv preprint arXiv:1610.00873*, 2016.

[3] J. Van Wonterghem, A. Alloum, J. Boutros, and M. Moeneclaey, "Performance comparison of short-length error-correcting codes," *arXiv preprint arXiv:1609.07907*, 2016.

[4] M. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on order statistics," *IEEE Trans. Inform. Theory*, vol. 41, no. 5, pp. 1379–1396, 1995.

[5] B. Dorsch, "A decoding algorithm for binary block codes and j-ary output channels," *IEEE Trans. Inform. Theory*, vol. 20, no. 3, pp. 391–394, May 1974.

[6] S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, E. Şaşoğlu, and R. Urbanke, "Reed-Muller codes achieve capacity on erasure channels," *IEEE Trans. Inform. Theory*, vol. 63, no. 7, pp. 4298–4316, 2017.

[7] T. R. Halford and K. M. Chugg, "Random redundant soft-in soft-out decoding of linear block codes," in *Proc. IEEE Int. Symp. Inform. Theory*. IEEE, 2006, pp. 2230–2234.

[8] T. Hehn, J. B. Huber, S. Laendner, and O. Milenkovic, "Multiple-bases belief-propagation for decoding of short block codes," in *Proc. IEEE Int. Symp. Inform. Theory*. IEEE, 2007, pp. 311–315.

[9] T. Hehn, J. B. Huber, O. Milenkovic, and S. Laendner, "Multiple-bases belief-propagation decoding of high-density cyclic codes," *IEEE Trans. Commun.*, vol. 58, no. 1, pp. 1–8, 2010.

[10] J. Jiang and K. R. Narayanan, "Iterative soft-input-soft-output decoding of Reed-Solomon codes by adapting the parity-check matrix," *IEEE Trans. Inform. Theory*, vol. 52, no. 8, pp. 3746–3756, 2006.

[11] A. Kothiyal, O. Y. Takeshita, W. Jin, and M. Fossorier, "Iterative reliability-based decoding of linear block codes with adaptive belief propagation," *IEEE Commun. Letters*, vol. 9, no. 12, pp. 1067–1069, Dec. 2005.

[12] M. H. Taghavi and P. H. Siegel, "Adaptive methods for linear programming decoding," *IEEE Trans. Inform. Theory*, vol. 54, no. 12, pp. 5396–5410, Dec. 2008.

[13] M. Bossert and F. Hergert, "Hard- and soft-decision decoding beyond the half minimum distance—An algorithm for linear codes," *IEEE Trans. Inform. Theory*, vol. 32, no. 5, pp. 709–714, Sept. 1986.

[14] T. J. Richardson and R. L. Urbanke, *Modern Coding Theory*. New York, NY: Cambridge University Press, 2008.

[15] D. Muller, "Application of Boolean algebra to switching circuit design and to error detection," *IRE Tran. on Electronic Computers*, vol. EC-3, no. 3, pp. 6–12, Sept 1954.

[16] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*. North-Holland, Amsterdam, 1977.

[17] J. Chen and M. P. C. Fossorier, "Density evolution for two improved BP-based decoding algorithms of LDPC codes," *IEEE Communications Letters*, vol. 6, no. 5, pp. 208–210, 2002.

[18] J. Feldman, M. J. Wainwright, and D. R. Karger, "Using linear programming to decode binary linear codes," *IEEE Trans. Inform. Theory*, vol. 51, no. 3, pp. 954–972, March 2005.

[19] S. Barman, X. Liu, S. C. Draper, and B. Recht, "Decomposition methods for large scale lp decoding," *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 7870–7886, 2013.