Channel Coding Strategies for 5G

Nima Mohammadi

Abstract

In this report, we aim to provide an introduction of the two state-of-the art channel encoding techniques proposed for the fifth generation wireless systems (5G), namely Low Density Parity Check (LDPC) and Polar codes. Although a thorough examination of this topic is precluded by the space limitations of this document, we hope to provide some preliminary description of these codes and the problems they aim to solve for 5G telecommunication system. We accompany our discussion with simulation results via the newly added 5G NR blocks in the communications toolbox of MATLAB.

1. Introduction

Channel coding, which is also known as forward error correction (FEC) refers to the process of detection and correction of error bits that may happen due to noise, interference or fading in a digital communication system. Channel encoding can be added to the communication system as modules that add extra redundant bits at the transmitter side (i.e. encoder) and then use those extra bits to correct possible errors at the receiving side and recover the original message (i.e. decoder). Hence, the redundancy added in the information message helps in increasing the reliability of the data received and also improves the fidelity of the received signal. Some of the popular channel encoders are Reed-Solomon codes, Low Density Parity Check (LDPC) codes, Turbo codes, and Convolution codes (Fig. 1.1). The channel encoded data is then passed to the channel modulator (Fig. 2.2). The reconstructed information message at the receiving side would then be an approximation of the original message, due to errors caused by the channel.

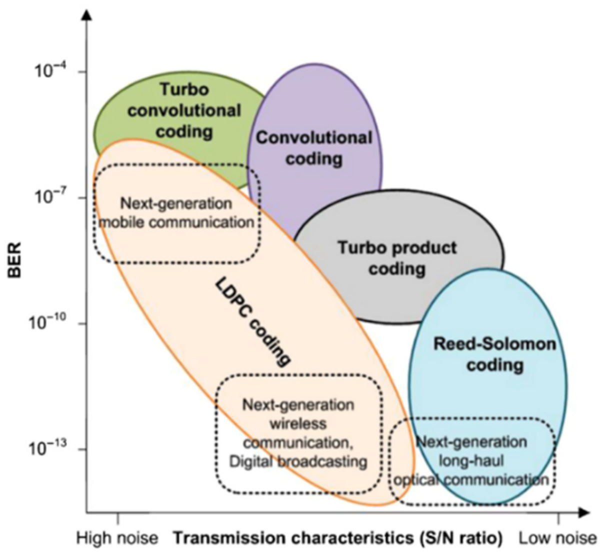


Figure 1.1. Comparison of famous channel encoding schemes

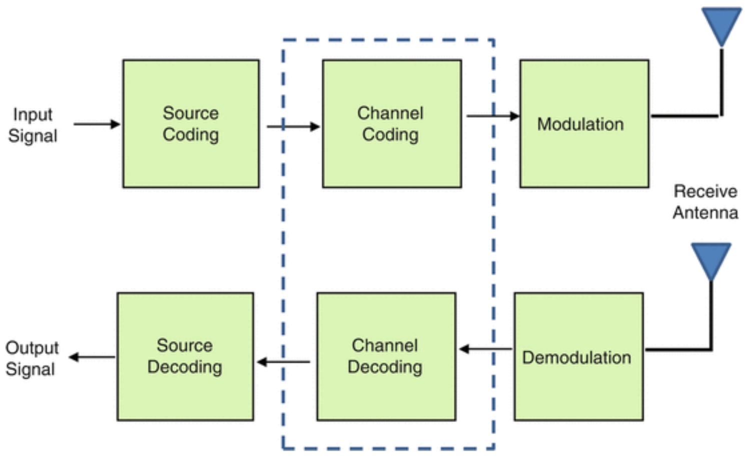


Figure 1.2. Schematic drawing of a communication system with channel coding [Faruque, 2016]

Depending on the mechanism of adding the redundancy to the messages, error correction codes are classified into Convolution codes and Block codes. The output of convolution codes depends on the current input as well as the previous inputs/outputs, hence the required memory for storing additional information for encoding. In contrast, the block coding schemes are memoryless, in the sense that the codewords are generated independently from each other. The information is processed in form of blocks. After this preliminary introduction it seems appropriate to mention about the subject matter of the report.

To see why in 5G it has been decided to move away from previous channel encoding choices in 4G to LDPC and Polar codes it seems reasonable to iterate the scenarios that have been defined for this generation. 5G introduces three new use cases:

- **Enhanced Mobile Broadband (eMBB)**: Bandwidth-driven use cases that require high data rates for wide mobile coverage.
- **Ultra-Reliable Low Latency Communications (URLLC)**: Requirements on latency and reliability of mission-critical applications such as autonomous vehicle (V2X), remote surgery/healthcare, or time-critical factory automation.
- **Massive Machine Type Communications (mMTC)**: To provide connectivity for a large number of devices that transmit intermittently low traffic, found in IoT devices.

With these scenarios at mind, it is clear that high-speed, low-latency, very reliable transmission are the goals at the heart of 5G for which it requires new high-performance forward error correction techniques to be introduced.

5G NR uses low density parity check (LDPC) codes for the data transmission for mobile broadband (MBB) services and polar codes for the control signaling. In that regard, LDPC replaces turbo codes for data channels and polar code replaces tail-biting convolutional codes for control channels.

LDPC codes are attractive from an implementation perspective, especially at multi-gigabits-per-second data rates. The LDPC codes considered for NR use a rate-compatible structure unlike the LDPC codes used in other wireless technologies. This permits the transmission at different code rates and for HARQ operation.

NR employs Polar Codes for the physical layer control signaling where the information blocks are relatively small compared to data transmission. By concatenating the polar code with an outer code and by performing successive cancellation list decoding, good performance is achieved at shorter block lengths. Reed-Muller codes are used for the smallest control payloads.

2. NR LDPC Coding

LDPC codes were first introduced by Gallager in his doctoral thesis [Gallager, 1962] but were then largely forgotten. They were rediscovered in the mid-1990s by several authors [MacKay and Neal, 1995, Alon and Luby, 1996]. A low-level analysis of how the

LDPC code works is beyond the scope of this report. As an intro to this code, assume we have the parity-check matrix (LDPC matrix) in Fig. 2.1. As the name implies the matrix is sparse and the number of 1s are much lower than the total number of elements. The complexity increases as the number of 1s goes up. This is not the case with our toy example, but we can assume it without loss of generality, for the actual larger sparse parity-check matrices. With each parity-check matrix we can come up with a bipartite graph called Tanner graph which is a very useful graphical representation of the matrix. Each column of the matrix is represented by a blue circle on the left (bit nodes) and similarly each row is represented by a red square on the right (check nodes). Edges indicate the existence of a one in the element of the corresponding row and column. Tanner graphs are especially very useful in describing the decoders. If we multiply this parity-check code with our code word, then $Hc^T = 0$. Then focusing on the second row, it corresponds to $c_2 + c_4 + c_6 + c_7 = 0$ which belongs to a (4, 3) Single Parity Check (SPC) code. That is every check node is enforcing a SPC on the bits it is connected to, forming a set of local constraints. We can see that based on this construct, LDPC is built on top of a series of "simpler" constructs, and decoding LDPC is a matter of decoding these SPC constraints and combining them in a clever way. In the standards, the LDPC almost always has a protograph construction, which describes how the parity check matrix can be built by expanding smaller base matrices via right shift permutation. In 5G-NR we have two base graphs and several expansions.

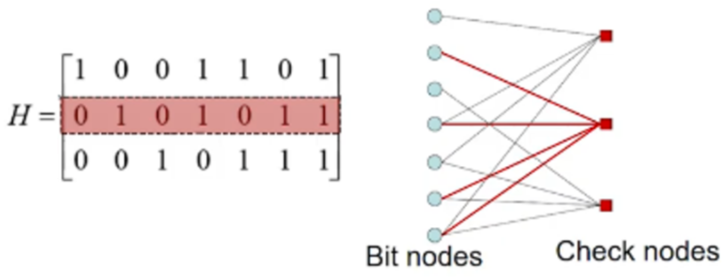

$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$
Bit nodes
Check nodes

Figure 2.1. An toy LDPC matrix with the corresponding Tanner graph

LDPC codes combined with soft decision iterative decoding provide currently the best achievable performance in digital communication systems [Tomlinson et al., 2017]. LDPC codes have attracted a lot of interest since then, including the incorporation in several IEEE standards such as IEEE 802.16e, IEEE 802.11n, and so on. Turbo and LDPC codes adopt coding schemes which are decoded by exchanging soft-decision information using iterative algorithms. As a consequence, bit error rate (BER) performance close to the Shannon limit can be achieved by these decoders. The hardware implementation of an LDPC decoder is relatively less complicated due to the inherent parallel nature of LDPC codes than that of Turbo decoders with comparable BER performance. Also because the interleaving is distributed in the code itself the LDPC decoder does not require the design of complex inter-leavers. All these variables make LDPC code suitable for wireless applications of the next generation that require high-performance encoders/decoders with low computational complexity and a low requirement for hardware resources.

The main advantages of 5G NR LDPC codes compared to turbo codes used in 4G LTE are:

- Improved area throughput efficiency and significantly better achievable peak throughput
- Decreased decoding complexity and enhanced decoding latency (particularly at high code rates) as for the possibility of higher degree of parallelization
- Enhanced performance, with error floors around or below the 10^{-5} BLER across all code sizes and rates.

These benefits make NR LDPC codes convenient to achieve high throughputs and make it suitable for ultra-reliable and low-latency requirement targeted by 5G with 20 Gb/s and 10 Gb/s targeted peak data rates for downlink and uplink, respectively.

5G LDPC codes adopt the structure of quasi-cyclic (QC) LDPC codes [Fossorier, 2004] which naturally enables parallelism in encoding and decoding, and high-throughput encoder and decoder can be realized by such parallelism. 5G channel codes for data should also support HARQ, and 5G LDPC codes are designed to efficiently support incremental redundancy (IR) HARQ. Such IR-HARQ design of 5G LDPC codes effectively reduces the size of encoding and decoding graph when the operating code rate is high, which also helps the realization of high throughput.

Other important features of 5G channel codes include the rate compatibility to select an arbitrary amount of transmitted bits from the mother code output and the variable code length. Figure **??** describes the 5G LDPC codes of an encoder/transmitter processing chain.

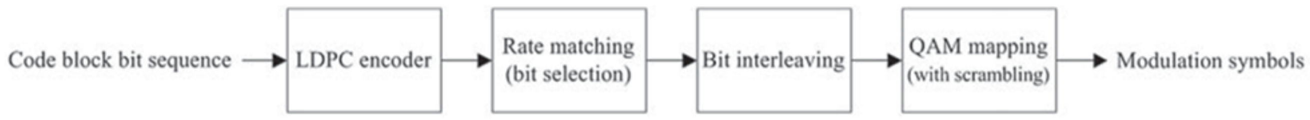


Figure 2.2. Schematic drawing of LDPC in 5G

NR LDPC codes use a quasi-cyclic structure, where the parity-check matrix (PCM) is defined by a smaller base matrix [Liva et al., 2008]. Each entry of the base matrix represents either a $Z \times Z$ zero matrix or a shifted $Z \times Z$ identity matrix, where a cyclic shift (given by a shift coefficient) to the right of each row is applied.

NR LDPC base matrix 1 is shown in Figure 2.3. The colored squares all correspond to shifted $Z \times Z$ identity matrices, while the white squares correspond to $Z \times Z$ zero matrices. Unlike the LDPC codes implemented in other wireless technologies, NR LDPC codes have a rate-compatible structure, similar to the codes proposed in [Chen et al., 2015]. Codewords of different rates can be generated by including a different number of parity bits, or equivalently, by using a smaller subset of the full PCM. The advantage of this structure is that for higher rates, the PCM, and, hence, the decoding complexity and latency, is smaller. This contrasts with the LTE turbo codes, which have constant decoding complexity and latency irrespective of the code rate.

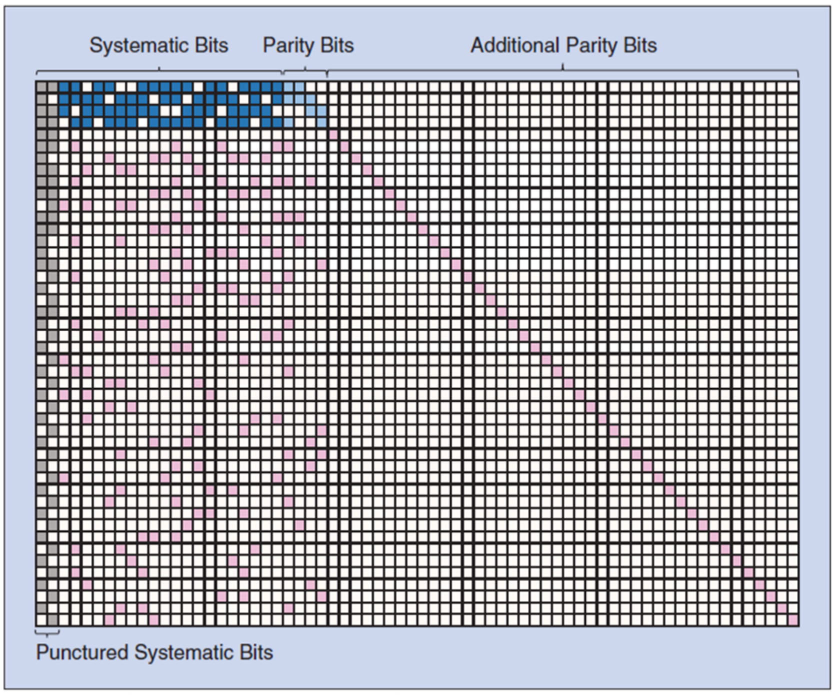


Figure 2.3. The structure of NR LDPC base matrix 1. Each square corresponds to one element in the base matrix or a $Z \times Z$ subblock in the PCM.

2.1. Two Base Matrices

The NR data channel supports two base matrices to ensure that good performance and decoding latency is achieved for the full range of code rates and information block sizes in NR. The parameters of base matrix 1 and base matrix 2 are given in Table I.

Base matrix 1, shown in Figure 2.3, is optimized for large information block sizes and high code rates. It is designed for a high maximum code rate of $8/9$ and may be used for

Parameter	Base Matrix 1	Base Matrix 2
Minimum design code rate	1/3	1/5
Base matrix size	46×68	42×52
Number of systematic columns Maximum information block size K	$8,448(= 22 \times 384)$	$3,840(= 10 \times 384)$
Number of nonzero elements	316	197

Table I NR LDPC base matrix parameters

code rates up to $R = 0.95$. Base matrix 2 is optimized for small information block sizes and lower code rates than base matrix 1. The lowest code rate for base matrix 2 without using repetition is $1/5$. For each base matrix, 51 PCMs are defined, each corresponding to a subblock size Z between two and 384, thus resulting in different ranges of information block size. Specifying PCMs for many different block sizes allows the code performance to be optimized in each case. In total, there are 102 PCMs defined for the NR data channel. For comparison, we note that IEEE 802.11n only specifies 12 PCMs with four different code rates and three different information block sizes.

The parameters for the two base matrices shown in Table I show a significant overlap both in information block size K (in bits) and in code rate R where both base matrix 1 and base matrix 2 may be used. For K and R in this region, the rules for selecting a base matrix based on information block size K and code rate R depend on the performance of the specific LDPC codes. In general, the base matrix with the best performance for a certain range of K and R is used.

From a decoding complexity point of view, for a given K value, it is beneficial to use base matrix 2, since it is more compact and utilizes a larger shift size Z (i.e., more parallelism) relative to base matrix 1. The decoding latency is proportional to the number of nonzero elements in the base matrix. Since base matrix 2 has much fewer nonzero elements than base matrix 1 for a given code rate (e.g., base matrix 2 contains 38% as many nonzero elements as base matrix 1 for code rate $1/3$), its decoding latency is significantly lower.

Figure 2.4 shows the regions of code rate and information block size K, for which base matrix 1 and base matrix 2 are used. In general, base matrix 2 is used for low code rates, and base matrix 1 is used for high code rates. For $K \leq 308$, only base matrix 2 is used since base matrix 2 performs better at all code rates in this range. For $308 \leq K \leq 3840$,

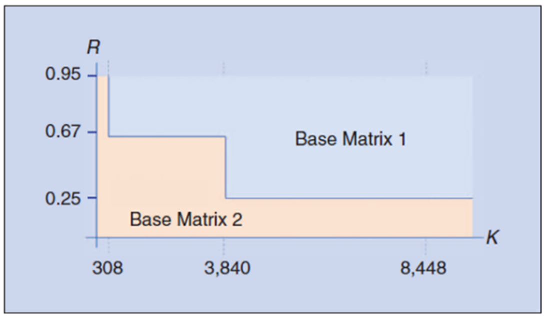


Figure 2.4. The usage of the two base matrices in 5G NR

Channel	AWGN
Modulation	QPSK
Code rates	$1/3, 2/5, 1/2$
Code	QC-LDPC
Information block lengths	2000,4000,8000
Decoding mechanism	Flooding BP

Table II Simulation parameters

base matrix 2 is used for code rates up to $2/3$ since base matrix 2 is optimized for $1/5 \leq R \leq 2/3$. Puncturing can be used to achieve higher code rates than 2/3 for base matrix 2, but base matrix 1 performs better in this range. For $K \geq 3840$ and $R \leq 1/4$, base matrix 2 is used because of its superior performance over base matrix 1, which requires the use of repetition to reach any code rate lower than $1/3$. The switch is not at a rate of $1/3$ because code block segmentation results in fewer code blocks when base matrix 1 is used as opposed to base matrix 2, which offsets the additional coding gain from base matrix 2 for rates between $1/4$ and $1/3$ when $K \geq 3840$.

2.2. A Performance Evaluation for eMBB Data Channel

This section mentions some performance evaluation results on a simulation that has been conducted in [Rao, 2019]. The utilized simulation parameters for the data channel can be found in Table II.

A uniform parity check matrix H is preferred for 5G eMBB channel coding [Rao, 2019]. The uniform base matrix points to the fact that code base matrix is derived from a uniform base matrix for any code block size or code rate. On each of the code blocks, QPSK modulation is applied and white Gaussian noise of various powers is added to the modulated bits. On the received noisy signal, QPSK demodulation is then performed. At last, the flooding belief propagation (BP) algorithm is applied for decoding the demodulated signal to recover the information bits and the BLER is calculated.

For QC-LDPC codes, the performance in terms of BLER when QPSK modulation is performed on AWGN channel for code rates 1/3, 2/5, and 1/2 with information block lengths K = 2000, 4000, and 8000 is depicted in Figure 2.5. The BLER values are computed by averaging over 100 independent trials.

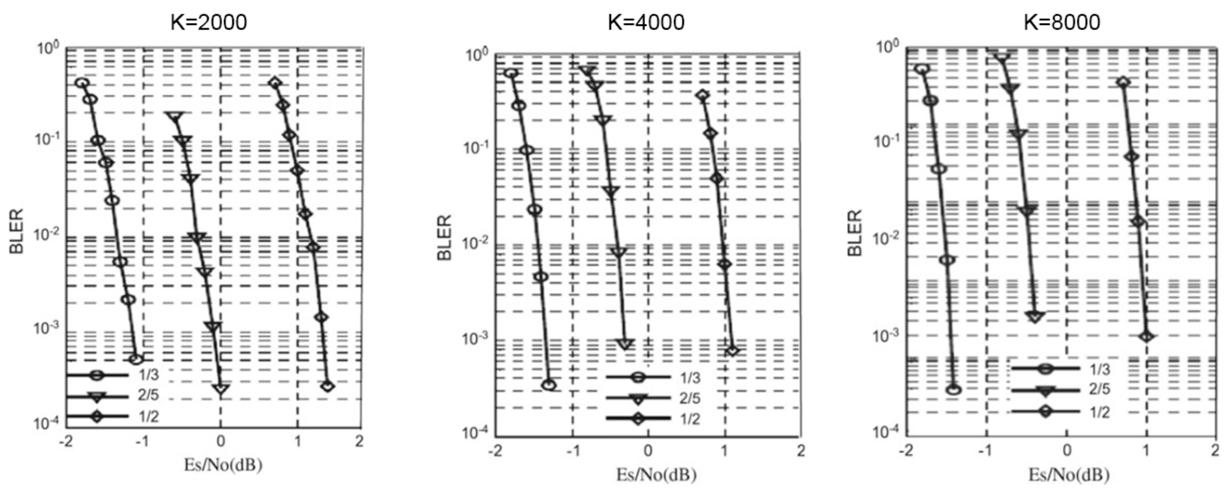


Figure 2.5. BLER performance of QC-LDPC codes with list decoding for $K = 2000$ (left), $K = 4000$ (middle) and $K = 8000$ (right)

3. Introduction to polar codes

Polar codes are a class of capacity-achieving codes introduced by Arikan [Arikan, 2009]. In the 5G standardization process of 3GPP, polar codes are adopted as channel coding for uplink and downlink control information for the eMBB communication service.

The construction of a polar code involves the identification of channel reliability values associated with each bit to be encoded. This identification can be effectively performed given a code length and a specific SNR. Within the 5G framework, however, various code lengths, rates and channel conditions are foreseen, and it is not feasible to have a different reliability vector for each parameter combination. Therefore, a substantial effort has been made to design polar codes that are easy to implement, with low description complexity, while maintaining good error-correction performance for multiple code and channel parameters.

3.1. Channel polarization

Introduced in [Arikan, 2009], the channel polarization phenomenon consists of a transformation that generates N synthetic bit-channels from N independent copies of a separate binary-input memoryless channel (B-DMC). In the sense that each of them can transmit a single bit at a different reliability, i.e. with a different probability of being correctly decoded, the new synthetic channels are then polarized. If N is sufficiently high, the synthetic channels' mutual information is either close to zero (completely noisy channels) or close to one (perfectly noiseless channels), resulting in extreme capacity channels.

The mathematical foundations of polar codes lie in the discovery of the polarization phenomenon of matrix $\mathrm{G}_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$, also known as basic polarization kemel. This matrix enables the encoding of a two-bits input vector $\mathbf{u} = [u_0, u_1]$ into code word $\mathrm{d} = [d_0, d_1]$ as $\mathrm{d} = \mathrm{u} \cdot \mathrm{G}_2$, namely having $d_0 = u_0 \oplus u_1$ and $d_1 = u_1$. The polarization effect brought by G_2 is more evident for binary erasure channels (BECs), where the transmitted bit is either received correctly or lost with probability δ. If the input bits are decoded sequentially, bit u_0 cannot be recovered as $u_0 = d_0 \oplus d_1$ if any of the code bits has not been received, hence with probability $\delta_0 = (2-\delta)\delta > \delta$. After u_0 has been correctly decoded, input bit u_1 can be decoded either as $u_1 = d_1$ or $u_1 = u_0 \oplus d_0$ hence it is sufficient that at least one of the two code bits has been received: consequently, failed decoding of this bit happens with probability $\delta_1 = \delta^2 < \delta$. As a result, input bit u_0 is transmitted over a degraded synthetic BEC with erasure probability $\delta_0 > \delta$, while u_1 is transmitted over an enhanced synthetic BEC with erasure probability $\delta_1 < \delta$

3.2. Code design

Polar codes are based on the concatenation of several basic polarization kernels, creating a cascade reaction that speeds up the synthetic channels polarization while limiting encoding and decoding complexity. This concatenation generates a channel transformation matrix defined by the n -fold Kronecker product of G_2, which can be recursively calculated as $\mathrm{G}_\mathrm{N} = \begin{bmatrix} \mathrm{G}_{\mathrm{N}/2} & 0 \\ \mathrm{G}_{\mathrm{N}/2} & \mathrm{G}_{\mathrm{N}/2} \end{bmatrix}$. As n approaches infinity, this construction creates channels that are either perfectly noiseless or completely noisy, for smaller values of n the synthetic channels polarization may be incomplete, generating intermediary channels that are only partially noisy. Figure 3.1 shows the bit-channels polarization enabled by the concatenation of $n = 10$ basic polarization kernels for a BEC with erasure probability $\delta = 1/2$.

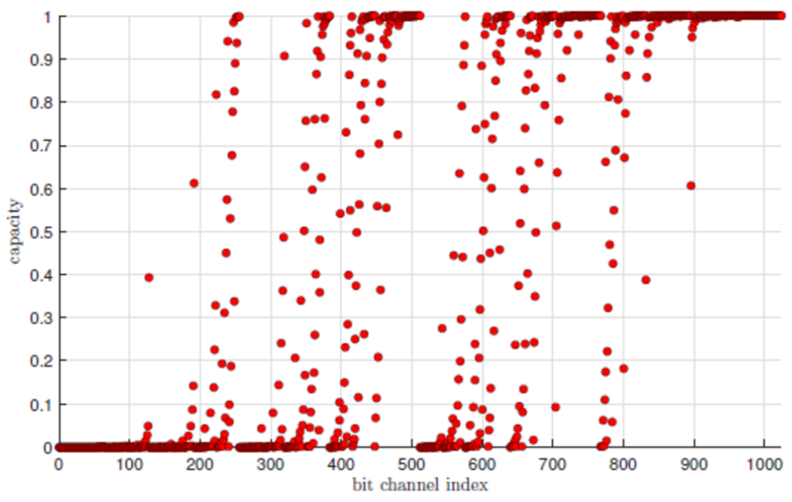


Figure 3.1. Virtual bit-channels capacities over a BEC 1=2

Polar codes allow only code lengths of powers of two, in the form of $N = 2^n$;, but on the other hand, the code dimension K can take an arbitrary value. The purpose of the (N, K) polar code architecture is to define and use the K best synthetic channels with the highest efficiency to transmit the data bits. The reliability calculation of each synthetic channel makes it possible to sort it in the order of reliability and allocate the K information bits to the most accurate channels whose indices form the code information set $\mathcal{I}$. The remaining $N - K$ indices form a frozen code range of $\mathcal{F} = \mathcal{I}^C$ and their respective channels do not carry information.

3.3. Encoding and Decoding

The polar code (N, K) is defined by its $\mathrm{GN} = \mathrm{G2}^{\otimes n}$ channel transformation matrix and its $\mathcal{I}$ information setting. The code generator matrix is defined by the $\mathrm{G_N}$ sub-matrix, consisting of the rows whose indices are in $\mathcal{I}$. The recursive structure of $\mathrm{G_N}$, allows the encoding complexity to be decreased by adding an auxiliary input vector $\mathbf{u}$ of the length N with $ui = 0$ if $I \in \mathcal{F}$, and storing the bits of information in the remaining entries. The codeword $\mathrm{d} = [d_0, d_1, \ldots, d_{N-1}]$ is then determined by $\mathbf{d} = \mathbf{u} \cdot \mathbf{G}_\mathrm{N}$. Employin parallelization that would result in a time complexity of $\mathcal{O}\left(\log_2(N)\right)$.

In [Arikan, 2009], a decoding algorithm, referred to as successive cancellation (SC), has been introduced which may be visualized as depth-first search of a binary tree where the leaf nodes are the N bits to be estimated, and soft information on the received code bits is input at the root node, with a decoding complexity of $\mathcal{O}\left(N \log_2(N)\right)$.

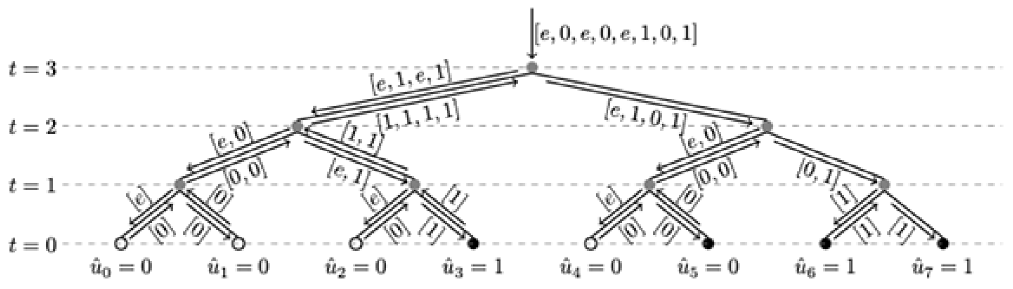


Figure 3.2. SC decoding of an (8, 4) polar code over a BEC; white and black dots represent frozen and information bits

Figure 3.2 shows the decoding tree of a $(8, 4)$ polar code with black leaf nodes representing information bits and white ones frozen bits. Taking a node at stage t as a reference, the message flow can be recursively described as shown in Figure 3.3. It uses 2^t soft inputs α received from its parent node to calculate 2^{t-1} soft outputs α^ℓ as $\alpha_i^\ell = f\left(\alpha_i, \alpha_{i+2^{t-1}}\right)$ to be transmitted to its left child. Then it will combine the 2^{t-1} hard decisions β^ℓ received from its left child with α to calculate 2^{t-1} soft outputs α^r for its right child as $\alpha_i^r = g\left(\alpha_i, \alpha_{i+2^{t-1}}, \beta_i^\ell\right)$. At the end, the 2^{t-1} hard decisions β^r received from its right child are combined with β^ℓ to calculate 2^t hard decisions β to be transmitted to its parent node as $\beta_i = \beta_i^\ell \oplus \beta_i^r$ if $i < 2^{t-1}$ and $\beta_i = \beta^r_{i-2^t-1}$ otherwise. When a leaf node is reached, the soft information is used to take a hard decision on the value of the information bits.

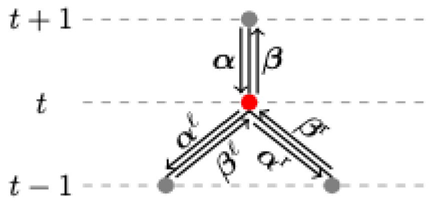


Figure 3.3. SC decoding node

3.4. Rate matching

As mentioned above, the polar code length N, by definition, is limited to powers of two, whereas it can have any number of bit channels K. This constrained setting is a limitation for usual 5G applications where the amount of information is fixed and a the length of the codeword is determined to achieve the desired rate. Hence, rate matching for polar codes is a length matching problem, and has been solved for 5G through classical coding theory methods like puncturing, shortening and extending [Richardson and Urbanke, 2008]

By not transmitting code bits in a predetermined pattern (i.e. a matching pattern), both

Channel	AWGN
Modulation	QPSK
Code rates	$1/6, 1/3, 2/5$ (downlink) 1/5 (uplink)
code	polar
Information block lengths	40,80,100 (down link) 200 (up link)
CRC length	19
Decoding mechanism	SCL
List size	8

Table III Simulation parameters

puncturing and shortening decrease the length of a mother code. The difference is in the meaning of the code bits belonging to the matching pattern. When the code bits are punctured, they are discarded on the decoder, whereas shortening adds a subscode that imposes the untransmitted code bits to have a fixed value that is already known by the decoder. For polar codes, shorting has been adopted for higher rate and puncturing is performed at low rate [Bioglio et al., 2017].

3.5. A Performance Evaluation for eMBB Data Channel

This section mentions some performance evaluation results on a simulation that has been conducted in [Rao, 2019]. The utilized simulation parameters for the data channel can be found in Table III.

In this simulation, cyclic redundancy code bits are appended to K randomly generated information bits for every frame. To obtain a code of length N, the information bits with the appended bits are encoded. The puncturing is then carried out and the M bits to be transmitted are acquired. M is determined by dividing K by R. The QPSK modulation is then applied on each of the M bits and the modulated bits are perturbed with white Gaussian noise of different powers. At the receiving side, QPSK demodulation is performed and LLR values are obtained (for punctured bits they are set to zero). The CRC aided SCL algorithm is then applied on the padded LLR values and K information bits are. The BLERs of this setting for the code rates $1/6$, $1/3$, $2/5$, and $1/2$ with information block lengths $K = 40$, 80, 100, and 200 is shown in Figure 3.4.

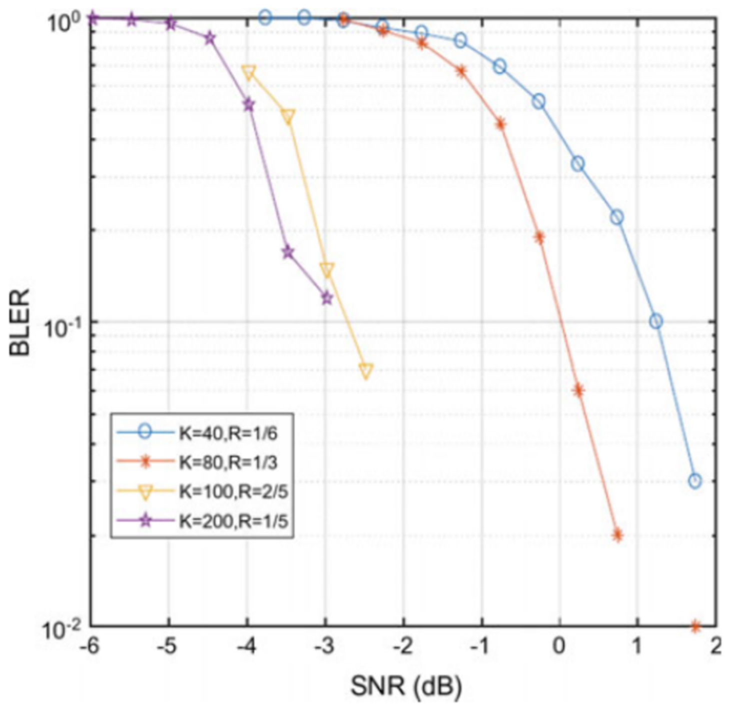


Figure 3.4. BLER performance for code rate 1/6, 1/3, 2/5, 1/5 for $K = 40$,80, 100, 200

4. MATLAB Simulations

Finally, we briefly provide codes that show how empirical performance analysis of these two channel encoding schemes can be calculated via MATLAB.

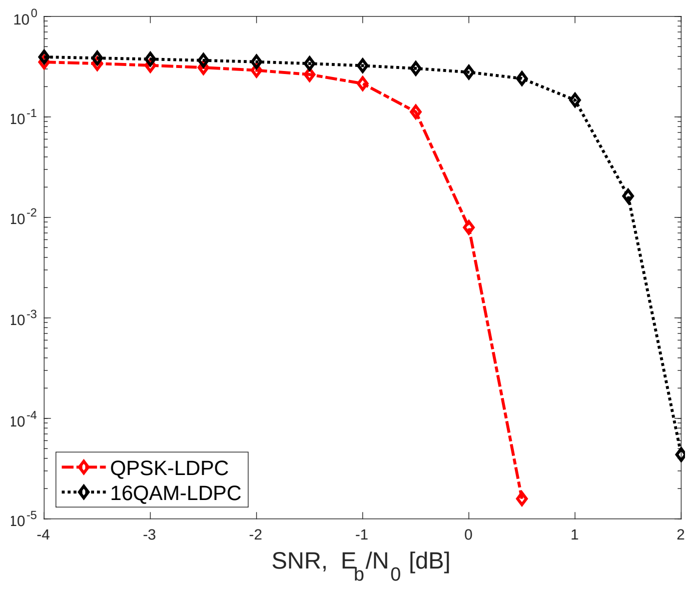
4.1. LDPC

Via simulations I found that 25 iterations seems to an adequate number for the LDPC decoder and not much gain was achieved with more iterations given the more time it takes to decode. I could not find any sample codes for these very new functions, nrLDPCEncode and nrLDPCDecode, but I believe my implementation should be correct. I have compared two modulation schemes, namely QPSK and 16QAM with LDPC channel encoding through an AWGN channel. The associated SNR vs BER curves are depicted in Figure 4.1.

```
clear;
clc;

```

```
rng(11);

numFrames = 200;

K = 2560;
F = 36;
R = 1/5;

bps_qpsk = 2;
bps_16qam = 4;

EbN0Vec = -4:.5:4;

EsNoVec_qpsk = EbN0Vec + 10*log10(bps_qpsk);
snrdBVec_qpsk = EsNoVec_qpsk + 10*log10(R);
noiseVar_qpsk = 1./(10.^(snrdBVec_qpsk/10));

EsNoVec_16qam = EbN0Vec + 10*log10(bps_16qam);
snrdBVec_16qam = EsNoVec_16qam + 10*log10(R);
noiseVar_16qam = 1./(10.^(snrdBVec_16qam/10));

BERs_16qam = zeros([length(EbN0Vec), 1]);
BERs_qpsk = zeros([length(EbN0Vec), 1]);

for k = 1:length(EbN0Vec)
    berr_qpsk = 0;
    berr_16qam = 0;
    chan_qpsk = comm.AWGNChannel('NoiseMethod', 'Variance', 'Variance',
        noiseVar_qpsk(k));
    chan_16qam = comm.AWGNChannel('NoiseMethod', 'Variance', 'Variance',
        noiseVar_16qam(k));
    for ii = 1:numFrames
        txcbs = randi([0 1], K-F, 1);
        fillers = zeros(F,1);
        txcbs = [txcbs;fillers];

        bgn = 2;
        txcodedcbs = nrLDPCEncode(txcbs,bgn);

        modOut_16qam = nrSymbolModulate(txcodedcbs,'16QAM');
        rSig_16qam = chan_16qam(modOut_16qam);
        rxLLR_16qam = nrSymbolDemodulate(rSig_16qam,'16QAM', noiseVar_16qam(k));

        [rxcbs_16qam,actualniters_16qam] = nrLDPCDecode(rxLLR_16qam, bgn, 20);

        berr_16qam = berr_16qam + biterr(txcbs(1:K-F), rxcbs_16qam(1:K-F));
```

Figure 4.1. SNR vs BER for QPSK-LDPC and 16QAM-LDPC



```matlab

        modOut_qpsk = nrSymbolModulate(txcodedcbs,'QPSK');
        rSig_qpsk = chan_qpsk(modOut_qpsk);
        rxLLR_qpsk = nrSymbolDemodulate(rSig_qpsk,'QPSK', noiseVar_qpsk(k));

        [rxcbs_qpsk,actualniters_qpsk] = nrLDPCDecode(rxLLR_qpsk, bgn, 20);

        berr_qpsk = berr_qpsk + biterr(txcbs(1:K-F), rxcbs_qpsk(1:K-F));
    end
    berr_16qam = berr_16qam / ((K-F) * numFrames);
    BERs_16qam(k) = berr_16qam;
    berr_qpsk = berr_qpsk / ((K-F) * numFrames);
    BERs_qpsk(k) = berr_qpsk;

    fprintf("EbN0: %d\tBER_qpsk: %.4f\tBER_16qam: %.4f\n",...
        EbN0Vec(k), berr_qpsk, berr_16qam);
end
%% Plots
fig1 = semilogy(EbN0Vec, BERs_qpsk, '-.rd', EbN0Vec, BERs_16qam, ':kd');
set(fig1, 'Linewidth', 2);
legend('QPSK-LDPC', '16QAM-LDPC', 'FontSize', 14, 'location', 'southwest');
xlabel('SNR,  E_b/N_0 [dB]', 'Fontsize', 16)
ylabel('BER', 'FontSize',16)
```

4.2. Polar codes

In this section we provide the MATLAB code for CRC-Aided Polar (CA-Polar) coding scheme with rate matching. The code below simulates a polar-coded QPSK-modulated link over AWGN. It has a better (or comparable) performance to LDPC and turbo codes and outperforms the tail-biting convolutional codes of LTE. Interleaving is specific to the downlink. The output length E and the information length K parameterize this code.

```
K = 54;
E = 128;

L = 8;
numFrames = 1000;
linkDir = 'UL';     % Link direction


if strcmpi(linkDir,'DL')
    % Downlink scenario (K >= 36, including CRC bits)
    crcLen = 24;
    poly = '24C';
    nPC = 0;
    nMax = 9;
    iIL = true;
    iBIL = false;
else
    % Uplink scenario (K > 30, including CRC bits)
    crcLen = 11;
    poly = '11';
    nPC = 0;
    nMax = 10;
    iIL = false;
    iBIL = true;
end

R = K/E;

EbN0Vec = -4:4;
bps = 2;
EsNoVec = EbN0Vec + 10*log10(bps);
snrdBVec = EsNoVec + 10*log10(R);
noiseVar = 1./(10.^(snrdBVec/10));

BERs = zeros([length(EbN0Vec), 1]);
BLERs = zeros([length(EbN0Vec), 1]);

```

```
for k = 1:length(EbN0Vec)
    chan = comm.AWGNChannel('NoiseMethod', 'Variance', 'Variance', noiseVar(k));
    numferr = 0;
    berr = 0;
    for i = 1:numFrames
        msg = randi([0 1], K-crcLen,1);

        msgcrc = nrCRCEncode(msg,poly);

        encOut = nrPolarEncode(msgcrc,E,nMax,iIL);
        N = length(encOut);

        modIn = nrRateMatchPolar(encOut,K,E,iBIL);

        modOut = nrSymbolModulate(modIn,'QPSK');

        rSig = chan(modOut);

        rxLLR = nrSymbolDemodulate(rSig,'QPSK', noiseVar(k));

        decIn = nrRateRecoverPolar(rxLLR,K,N,iBIL);

        decBits = nrPolarDecode(decIn,K,E,L,nMax,iIL,crcLen);

        berr = berr + biterr(double(decBits(1:K-crcLen)), msg);
        numferr = numferr + any(decBits(1:K-crcLen)~=msg);
    end
    berr = berr / ((K-crcLen) * numFrames);
    numferr = numferr / numFrames;
    BERs(k) = berr;
    BLERs(k) = numferr;
    fprintf("EbN0: %d\tBER: %.4f\tBLER: %.4f\n", EbN0Vec(k), berr, numferr);
end
%% Plots
subplot(1,2,1)
fig1 = semilogy(EbN0Vec, BERs, '-.rd');
set(fig1, 'Linewidth', 2);
legend('Polar Code')
xlabel('SNR,  E_b/N_0 [dB]', 'Fontsize', 14)
ylabel('BER', 'FontSize', 14)
title(sprintf('BER Polar Code (%s-Link)', linkDir), 'FontSize',14)
subplot(1,2,2)
fig2 = semilogy(EbN0Vec, BLERs, ':ks');
set(fig2, 'Linewidth', 2);
legend('Polar Code')
xlabel('SNR,  E_b/N_0 [dB]', 'Fontsize', 14)
```

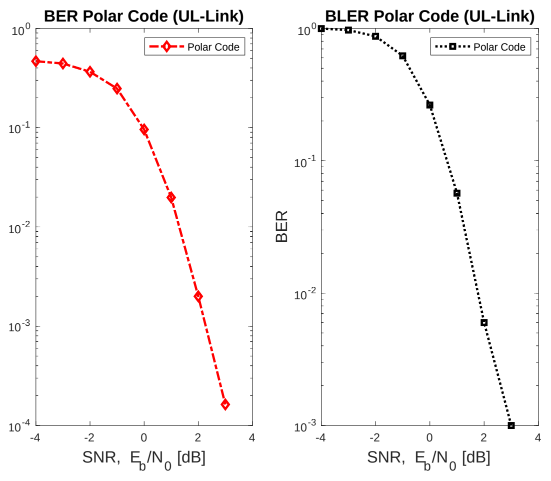


Figure 4.2. Polar - Uplink

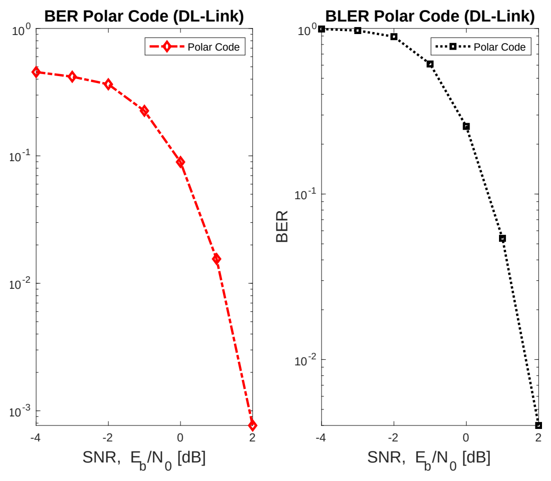


Figure 4.3. Polar - Downlink

```
ylabel('BER', 'FontSize', 14)
title(sprintf('BLER Polar Code (%s-Link)', linkDir), 'FontSize',14)
```

5. Concluding Remarks

In this report we presented an introduction of channel encoding and its necessity for communication systems. With a gentle preamble on the fifth generation of wireless systems and its requirements we provided a description of the two main channel encoding schemes that have been adopted into 5G, namely LDPC and Polar code, elaborating on their differences and use cases. At last, we presented our simulation codes using the newly added, but underdocumented, 5G NR capabilities in MATLAB.

References

[Alon and Luby, 1996] Alon, N. and Luby, M. (1996). A linear time erasure-resilient code with nearly optimal recovery. *IEEE Transactions on Information Theory*, 42(6):1732–1736.

[Arikan, 2009] Arikan, E. (2009). Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on information Theory*, 55(7):3051–3073.

[Bioglio et al., 2017] Bioglio, V., Gabry, F., and Land, I. (2017). Low-complexity puncturing and shortening of polar codes. In *2017 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pages 1–6. IEEE.

[Chen et al., 2015] Chen, T.-Y., Vakilinia, K., Divsalar, D., and Wesel, R. D. (2015). Protograph-based raptor-like ldpc codes. *IEEE Transactions on Communications*, 63(5):1522–1532.

[Faruque, 2016] Faruque, S. (2016). Introduction to channel coding. In *Radio Frequency Channel Coding Made Easy*, pages 1–16. Springer.

[Fossorier, 2004] Fossorier, M. P. (2004). Quasicyclic low-density parity-check codes from circulant permutation matrices. *IEEE transactions on information theory*, 50(8):1788–1793.

[Gallager, 1962] Gallager, R. (1962). Low-density parity-check codes. *IRE Transactions on information theory*, 8(1):21–28.

[Liva et al., 2008] Liva, G., Ryan, W. E., and Chiani, M. (2008). Quasi-cyclic generalized ldpc codes with low error floors. *IEEE Transactions on Communications*, 56(1):49–57.

[MacKay and Neal, 1995] MacKay, D. J. and Neal, R. M. (1995). Good codes based on very sparse matrices. In *IMA International Conference on Cryptography and Coding*, pages 100–111. Springer.

[Rao, 2019] Rao, K. D. (2019). Channel codes evolution for 5g. In *Channel Coding Techniques for Wireless Communications*, pages 465–476. Springer.

[Richardson and Urbanke, 2008] Richardson, T. and Urbanke, R. (2008). *Modern coding theory*. Cambridge university press.

[Tomlinson et al., 2017] Tomlinson, M., Tjhai, C. J., Ambroze, M. A., Ahmed, M., and Jibril, M. (2017). Ldpc codes. In *Error-Correction Coding and Decoding*, pages 315–354. Springer.