POLITECNICO MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

Executive Summary of the Thesis

Convolutional codes for short-packet communication over noncoherent channels

Laurea Magistrale in Telecommunication Engineering - Ingegneria delle Telecomunicazioni

Author: Matteo Ferro

Advisor: Prof. Maurizio Magarini

Co-advisors: Gianluigi Liva, Riccardo Schiavone

Academic year: 2023-2024

1. Introduction

Numerous emerging wireless applications rely on the sporadic transmission of short data units. These include for example machine-type communications, smart metering networks, the Internet of Things, remote command links and messaging services. For short information blocks, convolutional codes have been shown to approach the performance bounds of the best possible codes in coherent communications. However, for short packets, synchronization using known symbols (pilot symbols) introduces significant overhead. To address this issue, this Master's thesis analyzes convolutional codes over noncoherent channels, exploring the use of noncoherent receivers that eliminate the need for pilot symbols. We investigate this approach with both BPSK and QPSK modulation schemes, evaluating performance for zero-tail and tail-biting convolutional codes across different memory values. Finally, we assess the complexity and performance of the proposed solutions, comparing them with known approximations of the finite-length bounds for the best possible codes under coherent channels.

2. Convolutional codes

A binary convolutional code (CC) [1] receives at each time instant t a vector $\mathbf{u}_t$ of k bits and outputs a vector $\mathbf{x}_t$ of n bits, such that it is said to have *rate* $R = \frac{k}{n}$ (we will fix $n = 2,\ k = 1$, thus $R = \frac{1}{2}$).
The *memory* of the CC is specified by the parameter ν, which also describes the error correction capabilities of the code itself: a longer memory allows for a greater error correction power, which is paid by means of increased complexity. When we make use of a convolutional encoder, it could be important that, at some point, once we run out of information bits to transmit, we terminate the ongoing encoding process. For this discussion we focused on two alternatives:

- **Zero-tail** termination.
  The initial state is the all zero state and after K clock times the code is reset back to the starting state.
- **Tailbiting** termination.
  Initial state and ending state coincide, but there is no constraint on them being the all zero state.

When a word is received, the maximum likelihood (ML) criterion decides in favour of the word that maximizes the likelihood $p(\mathbf{y}|\mathbf{x})$:

$$\hat{\mathbf{x}}_{ML} = \arg\max_{\mathbf{x}\in\mathcal{C}} p(\mathbf{y}|\mathbf{x}). \tag{1}$$

By developing the log-likelihood function over the additive white Gaussian noise (AWGN) channel we end up with:

$$\hat{\mathbf{x}}_{ML} = \arg\min_{\mathbf{x}\in\mathcal{C}} d_E^2(\mathbf{y},\mathbf{x}) \tag{2}$$

$$d_E^2(\mathbf{y},\mathbf{x}) = ||\mathbf{y}-\mathbf{x}||_2^2 = \sum_{i=1}^{N}(y_i - x_i)^2 \tag{3}$$

where N is the length of the codeword $\mathbf{c}$ and $\mathbf{x}$ its BPSK-mapped version; $d_E^2(\mathbf{y},\mathbf{x})$ is the squared Euclidean distance between $\mathbf{y}$ and $\mathbf{x}$.
Ideally, the search has to be performed over all possible codewords belonging to the codebook but that is clearly too computationally heavy and time consuming. An efficient way to do so without resorting to any approximation is through the **Viterbi algorithm (VA)** [5], which makes use of two different metrics:

- We denote as Λ_t^i the **cumulative metric**, which corresponds to the minimum squared Euclidean distance amongst all possible paths reaching state S_i at time t.
- We define then $\lambda_t^{i,j}$ the **branch metric** at time t for the state transition between state S_i and S_j.

The resulting update rule is the following:

$$\Lambda_t^j = \min_i(\Lambda_{t-1}^i + \lambda_t^{i,j}).$$

The algorithm scans the code trellis diagram and every time two paths join in the same state it selects the one with the best metric and ignores the others. Thanks to the property of the squared Euclidean distance we can be sure that the paths cancelled are worse than the ones we keep. At the last time instant, the remaining path is the one associated to the ML codeword.

2.1. Wrap-around Viterbi algorithm

If the code is instead tailbiting, to ensure efficiency we have to employ a slightly modified version of the algorithm, such as the **wrap-around Viterbi algorithm (WAVA)**, that works as follows: while scanning the trellis we need to step by step keep track of the starting state of the survivor paths: once we reach the end, if the path that presents the overall best metric is not tailbiting, we restart from the beginning, setting the initial cumulative metric at each state equal to the value of the cumulative metric at the same state, computed at the end of the first iteration. We continue until we either get a tailbiting word or we reach a maximum number of iterations that was previously set.

3. Noncoherent channel

To better characterize the performance of CCs in short-packet communication we considered a noncoherent AWGN channel model, where the transmitted word is rotated by a (random) phase that is constant over the whole code block. This model captures, for instance, bursty transmissions of short packets in satellite links.
This means that the received signal has the same power of the original one but is rotated of a phase ϕ, which is set to be uniformly distributed between $-\pi$ and π. The channel model yields:

$$y_i = hx_i + n_i \tag{4}$$

where $h = e^{j\phi}$ and $\phi \sim U[-\pi,\pi)$. Since we considered a BPSK modulation, $x_i(c_i) \in \{-1,1\}$. Furthermore, the noise component is a circular symmetric complex Gaussian random variable with variance $2\sigma^2$, i.e. $n_i \sim \mathcal{CN}(0,2\sigma^2)$.
The signal-to-noise ratio (SNR) is defined as $\frac{E_s}{N_0} = \frac{1}{2\sigma^2}$ where E_s is the energy of a modulated symbol and N_0 is the single-sided noise power spectral density.
As the packet length is constrained to be significantly short (e.g. 64 information bits), it is reasonable to assume that the channel does not change importantly during the transmission of a single packet, as the coherence time is much larger than the transmission time.

3.1. Pilot-assisted transmission

Considering a channel coefficient h, the likelihood function can be expressed as

$$\begin{aligned}\hat{\mathbf{x}}_{ML} &= \arg\max_{x\in\mathcal{C}} p(\mathbf{y}|\mathbf{x},h)\\ &= \arg\min_{x\in\mathcal{C}} d_E^2(\mathbf{y},h\mathbf{x}).\end{aligned} \tag{5}$$

In practical applications the channel is not known and we have to introduce a way to obtain an estimate of its coefficient h. To do so, a typical approach is to send, along with the information bits, a certain amount of symbols that are known both at the receiver and the transmitter, which are called *pilot symbols*. This clearly has a cost in terms of rate, especially when we are dealing with short block-lengths, and there is a trade-off between the number of pilot symbols and the accuracy of the phase rotation estimate. Denoting by $\mathbf{x}_p = (x_p^1, x_p^2, \ldots, x_p^L)$ the pilot sequence and $\mathbf{y}_p = (y_p^1, y_p^2, \ldots, y_p^L)$ its observation at the channel output, we compute the ML channel estimate as

$$\hat{h} = \frac{\mathbf{x}_p^H \mathbf{y}_p}{||\mathbf{x}_p||^2} = \frac{\sum_{i=1}^{L} y_p^i}{L}. \tag{6}$$

Once we have our channel estimate, we can run the VA with the following metric

$$\lambda_t^{i,j} = d_E^2(\hat{h}^* \mathbf{y}_t, \mathbf{x}_t^{i,j}). \tag{7}$$

3.2. ML decoder without channel state information (CSI)

Adding pilot symbols to accurately estimate the channel shifts the codeword error rate (CER) curve right, originating a gap with respect to the ideal case with perfect CSI. With this in mind, we looked for a way to avoid the need of such overhead and close the above mentioned gap.
To do so, it is necessary to understand the behaviour of the likelihood function of the channel whenever the CSI is not available.
The ML metric was derived starting again from the ML decision rule:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x} \in \mathcal{C}} p(\mathbf{y}|\mathbf{x}).$$

As the phase is unknown, we can rewrite the likelihood function adding the dependence on ϕ:

$$p(\mathbf{y}|\mathbf{x}) = \int_{-\pi}^{\pi} p(\mathbf{y}|\mathbf{x}, \phi) \cdot p(\phi) \mathrm{d}\phi \qquad p(\phi) = \frac{1}{2\pi}.$$

Once this has been defined, the maximization yields:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x} \in \mathcal{C}} I_0 \left(\frac{1}{\sigma^4} \left| \sum_{i=1}^{N} x_i y_i \right|^2 \right) \tag{8}$$

where I_0 is the modified Bessel function of the first kind of order 0. Since I_0 has the property of being both even and, if we constrain I_0 in the domain $[0, \infty)$, monotonically increasing, maximizing it equals maximizing its argument:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x} \in \mathcal{C}} \left| \sum_{i=1}^{N} x_i y_i \right|. \tag{9}$$

This metric is regarded as ***noncoherent correlation***, as it is made of the product between transmitted and received signal. Its computation does not require any kind of phase estimation (thus noncoherent). Its maximization though cannot be performed by means of the previously introduced Viterbi algorithm, considering that the calculation of the noncoherent correlation cannot be performed recursively.

3.3. Blind Viterbi decoding

Once the metric is defined, we can introduce a new modified version of the VA, tailored to the noncoherent correlation (NCC) metric [4]. In contrast to the coherent case, it shall be noted that the algorithm is not ML, i.e., it is not optimum in the sense of minimizing the block error probability. The new rules for selecting the best path are outlined next:

1. Compute the *branch metric*

$$\lambda_t^{i,j} = \langle \mathbf{x}_t^{i,j}, \mathbf{y}_t \rangle \tag{10}$$

   $\mathbf{x}_t^{i,j}$ being the BPSK output of the transition from state i to state j at time t.

2. *Select the best path*

$$\hat{i} = \arg\max_i (|\Lambda_{t-1}^i + \lambda_t^{i,j}|^2). \tag{11}$$

3. Update the *cumulative metric*

$$\Lambda_t^j = \Lambda_{t-1}^{\hat{i}} + \lambda_t^{\hat{i},j}. \tag{12}$$

We performed Monte Carlo simulations to assess the error rate of the decoding algorithms introduced, with results displayed in Figure 1. For simplicity's sake, we will only show, for now, outcomes for the [133, 171] zero-tail terminated convolutional code (ZTCC), which is the optimum in terms of minimum distance ($d_{\text{min}} = 10$), amongst all possible ones with memory $\nu = 6$.
As a reference, we considered a system that is supposed to have *perfect CSI* as it represents a lower bound for the error correction performance of the different solutions implemented.

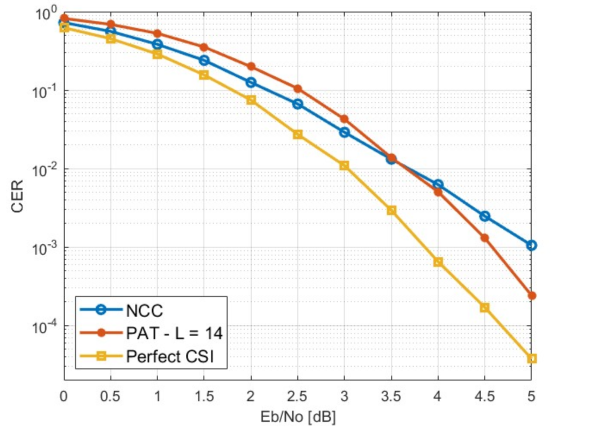

Figure 1: $K = 64$ information bits, NCC vs pilot-assisted transmission (PAT).

The pilot length $L = 14$ was selected as the most appropriate by running simulations for several values of L. The NCC algorithm behaves better than the PAT one at low values of $\frac{E_b}{N_0}$, where it benefits from the missing pilot overhead, but then seems to reach a floor which is due to the suboptimality of the metric update rule.

4. Viterbi tracking

To further improve previous results, it is fundamental to comprehend why and how errors happen and find a way to correct them. To do so, we collected wrongly decoded words and looked at the amount of flipped bits. Furthermore, we computed the ML channel estimate $\hat{h}$ as

$$\hat{h} = \frac{\langle \mathbf{y}, \hat{\mathbf{x}} \rangle}{||\mathbf{x}||^2} = \frac{\langle \mathbf{y}, \hat{\mathbf{x}} \rangle}{N}. \tag{13}$$

Moreover, we denoted the phase estimation error as $\Delta\phi = \angle\hat{h} - \angle h$.

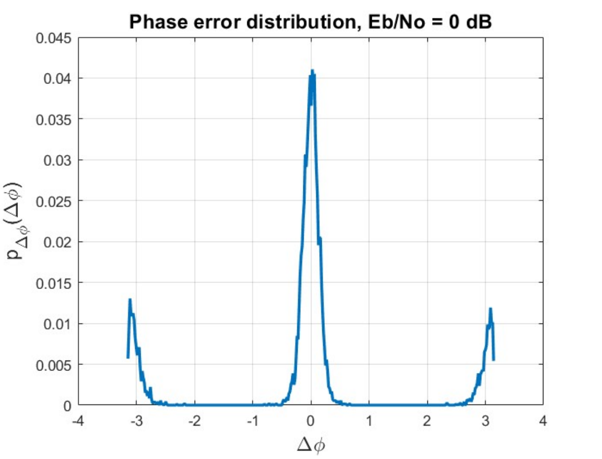

Figure 2: $\Delta\phi$ distribution, $\frac{E_b}{N_0} = 0$ dB.

When a decoding error happens, the phase estimated through the codeword is either very close to the actual one of the channel or approximately shifted by $\pm\pi$, as seen in Figure 2. This can be justified by observing that decoding errors lead to words that have Hamming distance close to $d_{\min}$ or $d_{\max}$ with respect to the correct one. We note that errors in the neighbourhood of $d_{\max}$ are a direct consequence of the pi-symmetry of the NCC update metric.

To exploit the way $\Delta\phi$ is distributed, we implemented a **Viterbi tracking algorithm (VTA)**: when an error is identified, it runs two different instances of the VA, once using the channel estimate $\hat{h}$ and once its rotated version $-\hat{h}$. Clearly the channel estimate will not be exact, but still it proves more than sufficient to yield a solid performance of the decoder. Once we have the two decoded words, we select the one that presents the highest likelihood, thus the best NCC metric $\Lambda = \langle \mathbf{y}, \hat{\mathbf{x}}_i \rangle$. If we suppose to have perfect knowledge of the decoding errors, we obtain the following:

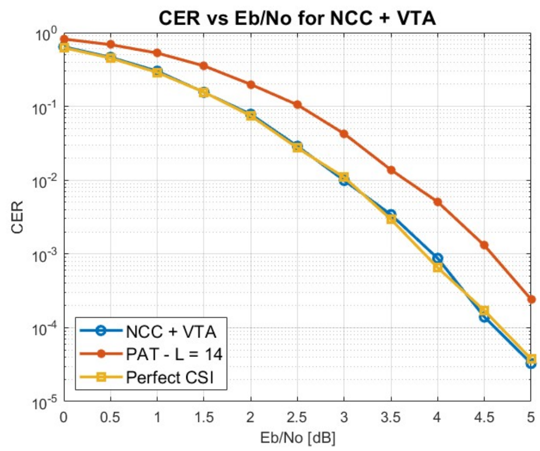

Figure 3: Extra loop with Viterbi tracking.

With this approach the NCC error rate approaches the performance of the perfect CSI decoder. This coding gain is paid in terms of complexity, as we need two more instances of the VA when an error is committed. This becomes less impacting at high $\frac{E_b}{N_0}$ values where the error rate is lower. In practical systems, we do not know when errors happen, thus we considered two different implementations to recognize decoding errors:

- Use **cyclic redundancy check (CRC) codes**.

- Set a **threshold** on the cumulative metric Λ below which the decoded word is supposed to be wrong.

4.1. CRC codes

Adding a CRC code [2] allows us to detect errors with great accuracy, provided that one with a sufficiently high order is used, but is paid through a loss in terms of rate. In short, a very simple algorithm is used to detect possible errors and it works by adding parity bits, though less than the pilot ones, at the end of the transmitted word. If the parity check performed at the receiver side yields a non-zero outcome, the codeword is supposed to be wrong and we enter the Viterbi tracking loop. As a further reference for our performance, we considered the normal approximation (NA) bound [3] for the AWGN channel, This is an approximation of the minimum achievable CER and it can be obtained under ML decoding of an ensemble of random codes at fixed rate and block-length.

4.2. Threshold mechanism

The second option examined is to set a threshold on the cumulative metric for the NCC algorithm, $\Lambda = \langle \mathbf{y}, \hat{\mathbf{x}} \rangle$, in order to perform the extra iteration step only in cases where the value of Λ is lower than Λ_{thr}.
We set empirically, looking at the Λ distribution, different values of percentage of errors on which the system performs the extra iteration step, namely 75%, 90% and 99%.

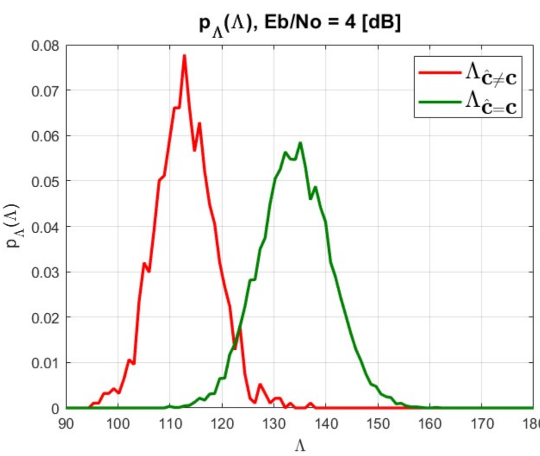

Figure 4: Distribution of $\Lambda = \langle \mathbf{y}, \hat{\mathbf{x}} \rangle$.

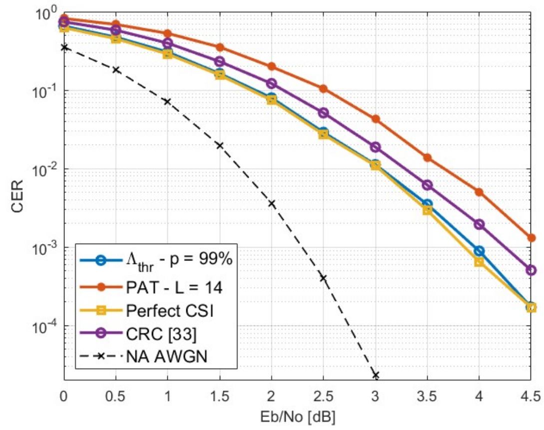

Figure 5: CER vs $\frac{E_b}{N_0}$ for CRC code of degree 4 and Λ_{thr} mechanism, ZTCC [133, 171].

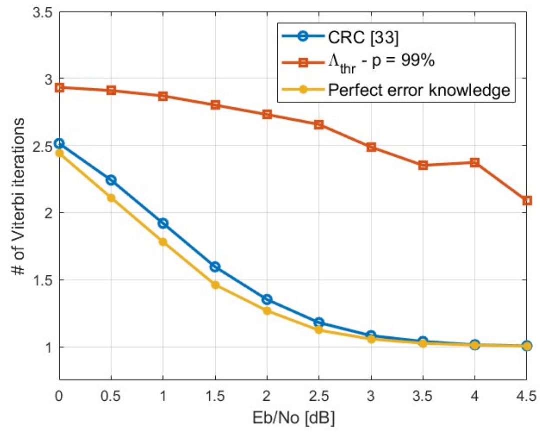

Figure 6: Average number of VA iterations.

If a sufficient amount of errors is identified after the first step, the decoder approaches the perfect CSI curve. This is paid in terms of complexity (Figure 6) and is due to the cumulative metric of wrong and correct words being for a good part superposed.
Overall the threshold approach shows a gain around 0.5 dB with respect to the PAT decoder, whilst the one exploiting a CRC code achieves a gain of 0.3 dB but with a lower complexity (Figure 5).

4.3. QPSK modulation

Exploiting the same approach, we were able to obtain satisfactory results with QPSK modulation as well, showing again a consistent improvement of approximately 0.3 dB with respect to the PAT decoder (Figure 7).

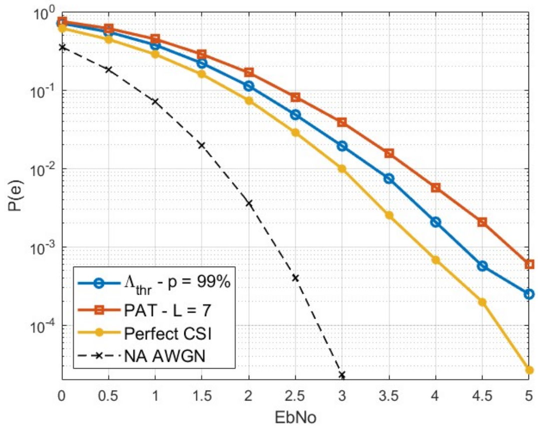

Figure 7: Λ_{thr} setup, ZTCC [133, 171], QPSK.

5. Tailbiting terminated codes

For further improvement, we tried using tailbiting convolutional codes (TBCCs), which do not require additional bits of termination. However, this solution encounters symmetry issues: some of the optimal CCs in terms of $d_{\min}$ present in their codebook the all ones codeword. This introduces a pi-symmetry, meaning that any codeword and its flipped version both belong to the codebook. This prevents the use of the ML decoder straight-forward, being the NCC metric equal for two distinct words. To solve this, we looked for the best non-symmetric code given a fixed memory value. To identify it, we developed new metrics which this time must take into account not only the code $d_{\min}$, but also its $d_{\max}$:

$$d_{best} = \min(d_{\min},\ N - d_{\max})$$
$$A_{best} = A(d_{best}) + A(N - d_{best})$$

A_{best} being the multiplicity of the lowest Hamming distance words according to this metric.

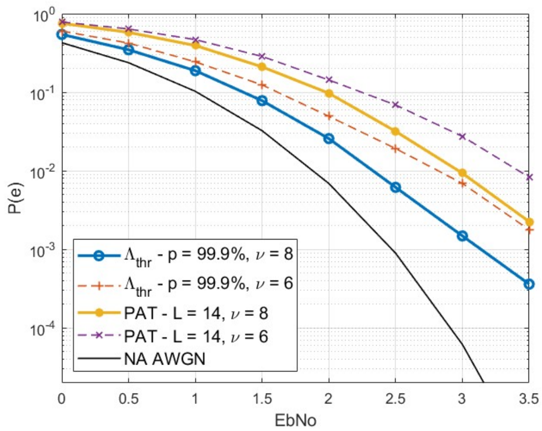

Figure 8: Λ_{thr} setup, TBCC [663, 711].

This time we show instead results for a memory 8 code (Figure 8). We note that the NCC memory 6 decoder is even better than the PAT one with memory 8. The average number of runs of the VA is just slightly shifted at different looping probabilities.
With a larger memory value, we are also able to significantly reduce the gap to the normal approximation bound, which is cut down to approximately 0.5 dB up to error rates of 10^{-3}

6. Conclusions

To conclude, we were able to demonstrate in different configurations the effectiveness of convolutional codes in the framework of short-packet communication. The advantages were proven at different SNR and error rate values and consisted in coding gains of approximately 0.3 to 0.5 dB.
All of this was shown for several different implementations and both BPSK and QPSK modulation schemes. When using memory 8 tailbiting codes it was also possible to get interestingly close to the NA bound. In every scenario, the coding gain was paid through a reasonable increase in complexity. However, it was possible to find satisfactory trade-offs that allowed for a restrained complexity while still achieving a significant gain with respect to the PAT decoder.

References

[1] Peter Elias. Coding for noisy channels. In *IRE WESCON Convention Record, 1955*, volume 2, pages 94–104, 1955.

[2] William Wesley Peterson and Daniel T Brown. Cyclic codes for error detection. *Proceedings of the IRE*, 49(1):228–235, 1961.

[3] Yury Polyanskiy, H Vincent Poor, and Sergio Verdú. Channel coding rate in the finite blocklength regime. *IEEE Transactions on Information Theory*, 56(5):2307–2359, 2010.

[4] Giorgio Taricco and Ezio Biglieri. Space-time decoding with imperfect channel estimation. *IEEE Transactions on Wireless Communications*, 4(4):1874–1888, 2005.

[5] Andrew Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, 13(2):260–269, 1967.