Adv. Radio Sci., 10, 159–165, 2012
www.adv-radio-sci.net/10/159/2012/
doi:10.5194/ars-10-159-2012


**Advances in Radio Science**

# Application of syndrome based Turbo decoding with adaptive computational complexity in LTE downlink

**J. Geldmacher**[1], **K. Hueske**[1], **M. Kosakowski**[2], **and J. Götze**[1]

[1]TU Dortmund University, Information Processing Lab, Otto-Hahn-Str. 4, 44227 Dortmund, Germany
[2]Research In Motion GmbH, Universitätsstr. 140, 44799 Bochum, Germany

*Correspondence to:* J. Geldmacher (jan.geldmacher@ieee.org)

**Abstract.** This paper describes the application of an adaptive complexity decoder for the Long Term Evolution (LTE) downlink Turbo code. The proposed decoding approach is based on the block syndrome decoding principle and enables adaptive reduction of decoding effort depending on current SNR and iteration number with negligible influence on decoding performance. Numerical results in context of LTE downlink using typical mobile channels are used to demonstrate the efficiency of the approach.

## 1 Introduction

Mobile communication usage has shifted from mainly speech-centered use to highly bandwidth demanding data communication: A mobile phone is no longer just a *phone*, but it has become a personal communication and multimedia device, which is also used for internet browsing, video and audio streaming, online-based navigation etc. The resulting increasing bandwidth demands are satisfied in one part by using more sophisticated signal processing in the physical layer, like MIMO transmission or advanced forward error correction (FEC) methods.

Considering the Long Term Evolution (LTE), one of the most complex parts of these baseband processing components is the FEC, which is realised using Turbo coding (Berrou et al., 1993). Turbo Codes are used in several applications, because of their excellent BER performance, compared to classical codes like convolutional or block codes. In case of ideal uncorrelated Gaussian inputs and very large code block lengths, Turbo Codes can achieve close-to-capacity BERs. However, good performance has to be payed for with high computational complexity: Turbo codes are decoded iteratively by using constituent decoders, which exchange their beliefs about the transmitted information bits in the form of Log Likelihood Ratios (LLRs). Typically the LogMAP or the MaxLog algorithm (Robertson et al., 1995) are used in the constituent decoders to generate these LLRs by doing forward and backward recursions on a trellis representation of the code. Traditionally this trellis represents the encoder $\mathbf{G}$ of the code.

In this paper, a different decoding approach is considered which is based on the so called syndrome former $\mathbf{H}^T$ of the underlying code. While this decoder is equivalent to the conventional decoder in terms of trellis complexity and decoding performance, it can be modified to achieve adaptive decoding complexity based on the so called Block Syndrome Decoding (BSD) approach. The BSD concept has been described earlier in context of Viterbi decoding (Geldmacher et al., 2009) and Turbo equalization (Geldmacher et al., 2010; Hueske et al., 2010). Applying it to Turbo decoding is, however, not straightforward as will be shown in this paper, but requires the extension of the Turbo decoder by means of a pre-correction of the constituent decoders inputs.

In the following Sect. 2 the BSD concept is summarized. The syndrome based, adaptive complexity Turbo decoder is explained in Sect. 3 and numerical results for typical LTE scenarios are shown in Sect. 4. Conclusions are drawn in Sect. 5.

## 2 Block syndrome decoding

In syndrome decoding, instead of the trellis of the encoder, the trellis of the syndrome former $\mathbf{H}^T$ is used, where each path represents an admissible error sequence $\boldsymbol{\epsilon}$. Given the hard decision $\mathbf{r}$ of a received distorted sequence, an admissible error sequence $\boldsymbol{\epsilon}$ is defined as any sequence that results in a valid code sequence, when applied to $\mathbf{r}$. This trellis can be constructed based on the syndrome former $\mathbf{H}^T$ of the code, and the syndrome sequence $\mathbf{b} = \mathbf{r}\mathbf{H}^T$ of $\mathbf{r}$ (Forney Jr., 1970; Schalkwijk and Vinck, 1976). Consequently, in this case a MAP decoder estimates the LLRs of the error in code or information bits, and this approach is referred to as syndrome

decoding. The resulting absolute LLR is identical for both approaches, and therefor the choice of the trellis has no influence on the decoding performance. Also trellis complexities of syndrome former and encoder are known to be identical in their minimal realizations (Forney Jr., 1970).

However, there is an interesting property of syndrome based decoding that can be exploited for reducing the decoding complexity. As the syndrome based decoder outputs LLRs of the error bits and not of the code bits, the probability of states and transitions depends on the number of errors in the decoders input sequence **r**: Less errors relate to a higher probability of the all-zero state in the trellis. This also means that if a subblock in the decoder's input sequence is error-free, then the zero-state is the most likely state in this subblock, and consequently that an estimation of error-free blocks can be interpreted as a state estimation.

Decoding complexity can then be reduced by identifying error-free subblocks before the decoding starts, and switching off the decoder for these subblocks. Only the remaining erroneous subblocks are actually processed by the decoder. In context of syndrome decoding, a very simple yet efficient criterion for estimating error-free subblocks is the syndrome **b** of the decoder's input sequence. Because of the orthogonality of the syndrome former $\mathbf{H}^T$ to the code, the syndrome only depends on the channel error $\boldsymbol{\epsilon_c}$,

$$\mathbf{b} = \mathbf{r}\mathbf{H}^T = (\mathbf{v}+\boldsymbol{\epsilon_c})\mathbf{H}^T = \mathbf{u}\mathbf{G}\mathbf{H}^T + \boldsymbol{\epsilon_c}\mathbf{H}^T = \boldsymbol{\epsilon_c}\mathbf{H}^T, \quad (1)$$

so that subsequences of a certain number of consecutive zeros in **b** indicate error-free subblocks in **r** with certain probability. In (1) $\mathbf{v} = \mathbf{u}\mathbf{G}$ denotes the code sequence corresponding to the information sequence **u**.

In summary, the basic idea of BSD consists of two steps:

1. *Preprocessing* Compute the syndrome **b** of the decoders input sequence and identify subblocks of a predefined minimum number of consecutive zeros. These subblocks are considered to be error-free.

2. *Decoding* Apply the syndrome-based decoder only to the remaining erroneous subblocks.

The required minimum number of consecutive zeros is a design parameter, which is called $\ell_{\min}$ in the following. It influences complexity reduction and loss of decoding performance: A smaller $\ell_{\min}$ will result in the detection of more error-free subblocks, i.e. more reduction of decoding complexity, but also has a higher chance of identifying a subblock as error-free which actually contains errors, i.e. higher degradation of decoding performance.

It is easy to see, that the BSD concept results in a decoding complexity that adapts to the number of errors contained in the decoders input sequence. In the worst case, if the input sequence contains many errors, no error-free subblocks are identified, and the decoder has to process the whole sequence. In this case the effort is equal to a conventional decoder plus the preprocessing overhead. However, if the decoder's input contains only few errors, more error-free subblocks can be identified and the decoding effort is reduced.

Considering a Turbo decoder, it is expected that the number of remaining errors in the a priori estimate of the systematic part decreases during Turbo iterations. Additionally, if an estimate of the parity part is employed to correct errors in the parity part, then the overall number of errors in the constituent decoder's input decreases with ongoing iterations and increasing SNR. Thus, employing the BSD concept yields complexity reduction with increasing SNR and ongoing iterations.

The necessity of correcting the constituent decoder's input sequence requires a modified transition metric, which accounts for the precorrection. Therefore the syndrome decoding approach for Turbo Codes described in this work differs from other syndrome based decoding algorithms (Schalkwijk and Vinck, 1976; Yamada et al., 1983; Tajima et al., 2002; Minowa, 2003; Geldmacher et al., 2010).

## 3 Adaptive complexity Turbo decoder

First the operation of the syndrome based constituent decoder is described (Sect. 3.1), then the Turbo decoding framework with precorrection and the BSD extension are explained (Sect. 3.2 and Sect. 3.3).

### 3.1 Syndrome based Turbo decoder

It is assumed, that a half rate systematic terminated encoder is employed, and that on receiver side depuncturing is realized by replacing punctured positions with zero reliability bits. One constituent decoder takes the received soft decision sequence $\tilde{\mathbf{r}}$, a precorrection sequence **x** and a priori LLRs of the errors $L_A(\boldsymbol{\epsilon})$ in the systematic part as input. For the systematic part, it generates an *a posteriori* hard decision estimate $\hat{\boldsymbol{\epsilon}}$ of the error in the harddecision **r** of $\tilde{\mathbf{r}}$ and extrinsic LLRs $L_E(\boldsymbol{\epsilon})$. The decoder performs the following steps:

1. *Precorrection and syndrome computation* The precorrection sequence **x** is applied to **r**, and the syndrome **b** is computed as

$$\mathbf{b} = (\mathbf{r} \oplus \mathbf{x})\mathbf{H}^T. \quad (2)$$

2. *Trellis operation* Given the trellis of $\mathbf{H}^T$ subject to **b**, the MaxLog algorithm is applied to generate an *a posteriori* estimate $L(\boldsymbol{\epsilon})$. The intial state for the forward recursion is set to the zero state, while the intial state for the backward recursion is selected according to the final state of the syndrome former from (2). The transition metric at time instant $t$ is given as

$$\tilde{x}_t^s \tilde{e}_{(p,q)}^s \left(|\tilde{r}_t^s| - L_A(\epsilon_t^s)/2\right) + \tilde{x}_t^p \tilde{e}_{(p,q)}^p |\tilde{r}_t^p|, \quad (3)$$

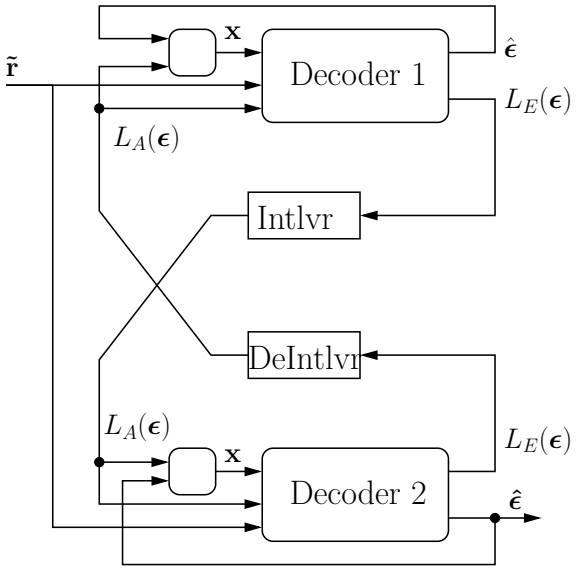


**Fig. 1.** Syndrome based Turbo decoder with precorrection.

where $\tilde{x}_t^s$, $\tilde{x}_t^p$ and $\tilde{e}_{(p,q)}^s$, $\tilde{e}_{(p,q)}^p$ represent BPSK modulated systematic and parity bits of precorrection sequence and error correponding to the trellis transition from states $p$ to $q$. This yields the a posteriori LLR $L(\mathbf{e})$ of the incremental error $\mathbf{e}$ w.r.t. $(\mathbf{r} \oplus \mathbf{x})$. Note that for LogMAP decoding proper normalization using the channel reliability would be required.

In order to generate the LLR $L(\boldsymbol{\epsilon})$ of the absolute error in $\mathbf{r}$, the sign of $L(\mathbf{e})$, which is an estimate of the error in $\mathbf{r} \oplus \mathbf{x}$, has to be flipped according to $\mathbf{x}$,

$$L(\epsilon_t^s) = -\tilde{x}_t^s L(e_t^s). \tag{4}$$

The harddecision $\hat{\boldsymbol{\epsilon}}$ of $L(\boldsymbol{\epsilon})$ delivers the *a posteriori* estimate of the channel error $\boldsymbol{\epsilon}_c$. Applying $\hat{\boldsymbol{\epsilon}}$ to the systematic part $\mathbf{r}^s$ of $\mathbf{r}$ results in the estimate $\hat{\mathbf{u}}$ of the original information bits $\mathbf{u}$.

3. *Generation of extrinsic LLR* The extrinsic LLRs $L_E(\boldsymbol{\epsilon})$, that are passed to the other constituent decoder, are generated by removing the a priori information $L_A(\boldsymbol{\epsilon})$ and the received systematic softbits from $L(\epsilon_t^s)$,

$$\begin{aligned} L_E(\epsilon_t^s) &= L(e_t^s) - (-\tilde{x}_t^s)\left(\tilde{x}_t^s |\tilde{r}_t^s| - \tilde{x}_t^s L_A(\epsilon_t^s)\right) \\ &= L(\epsilon_t^s) + |\tilde{r}_t^s| - L_A(\epsilon_t^s). \end{aligned} \tag{5}$$

### 3.2 Turbo decoder with precorrection

An overview of the syndrome based Turbo decoder with two constituent decoders is shown in Fig. 1. Like in the conventional Turbo decoder, both constituent decoders use the systematic and their according parity part of the received softbits, and, as a priori values, the interleaved extrinsic informations $L_E(\epsilon_t^s)$ generated by the other decoder. Additionally, the precorrection sequence $\mathbf{x}$ is used.

Generally an arbitrary sequence can be selected for $\mathbf{x}$, without changing the absolute values of the generated LLRs. However, for the BSD concept it is crucial, that the input $(\mathbf{r} \oplus \mathbf{x})$ of the syndrome former contains as few errors as possible, such that the hamming weight of $\mathbf{b}$ becomes small. This can be achieved by selecting $\mathbf{x}$ as an estimate of the channel error $\boldsymbol{\epsilon}_c$, which in context of the Turbo decoding framework can be done as follows:

- The systematic part $\mathbf{x}^s$ is set according to the hard decision of the a priori values $L_A(\boldsymbol{\epsilon})$.
- For the parity part $\mathbf{x}^p$, the output $\hat{\mathbf{u}}$ from the previous iteration is reencoded, which yields an estimated code sequence $\hat{\mathbf{v}}$. The comparison of the parity part $\hat{\mathbf{v}}^p$ of $\hat{\mathbf{v}}$ with $\mathbf{r}$, yields $\mathbf{x}^p = \mathbf{r}^p \oplus \hat{\mathbf{v}}^p$.

Thus the sequence $\mathbf{x}$ depends on both constituent decoders, such that it can be used as a measure for the decoding progress. In case of convergence of the decoder, it leads to a decreasing hamming weight of the syndrome sequence $\mathbf{b}$.

The described syndrome based Turbo decoder is identical to the conventional decoder in terms of decoding performance and trellis complexity. Additional effort is required to compute the sequence $\mathbf{x}$ and the syndrome $\mathbf{b}$. However, both are binary operations and can be considered of negligible complexity.

### 3.3 Block syndrome decoding of Turbo codes

Based on the described syndrome based Turbo decoder, the BSD concept (Geldmacher et al., 2009, 2010) as described in Sect. 2 can be applied to achieve a reduction of decoding effort. Because of the precorrection, the syndrome sequence $\mathbf{b}$ of a constituent decoder shows subsequences of consecutive zeros, that increase with ongoing iterations in case of convergence. This can be exploited to separate the input sequence into subblocks that are considered to be erroneous and subblocks that are considered to be error-free. Consequently, a reduction of decoding effort can be achieved by only processing the erroneous subblocks and neglecting the supposely error-free subblocks.

More precisely the syndrome based Turbo decoding algorithm from Sect. 3.1 is extended as follows:

1. *Preprocessing of syndrome* Identify subsequences of $\geq \ell_{\min}$ zeros in $\mathbf{b}$. Consider the corresponding subblocks in $\tilde{\mathbf{r}}$, except a padding of $\lfloor \ell_{\min}/2 \rfloor$ at the beginning and end of each subblock, as error-free, and the remaining subblocks as erroneous.

2. *Processing of subblocks*

   (a) *Erroneous blocks* The erroneous subblocks are processed by the syndrome based MAP decoder, which generates extrinsic values $L_E(\epsilon_t^s)$ and the estimated error $\epsilon_t^s$ for all $t$ in these subblocks. Note that these blocks can be considered to be terminated in the zero state, because the zero state is the most likely state in the preceding and succeeding error-free blocks.

   (b) *Error-free blocks* No processing is required for the supposedly error-free blocks. Instead, the extrinsic LLR is set to a sufficiently large value, $L_E(\epsilon_t^s) = \tilde{x}_t^s c$, with $c > 0$, and the estimated error is set to the systematic part of the pre-correction sequence, $\epsilon_t^s = x_t^s$. A reasonable choice for $c$ is the largest value in the quantization range of the LLR values.

The choice of the design parameter $\ell_{\min}$ affects the achievable reduction of decoding effort and possible loss in decoding performance. Given an acceptable loss in block error rate (BLER), it may be selected heuristically.

## 4 Simulation results

To show the efficiency of the proposed BSD Turbo decoder, it is analyzed in context of an LTE downlink scenario (Mehlführer et al., 2009). For the simulations presented in this section, the modulation and coding scheme (MCS) is set according to the Channel Quality Indicator (CQI). A single-user, single antenna scenario is considered, where the carrier frequency is set to $f_C = 2.1\,\text{GHz}$ and Doppler frequency is $f_{D,\max} = 0\,\text{Hz}$. For all simulations independent channel realizations are used. The bandwidth is set to 5 MHz for all simulations, and no control channel information is transmitted by the eNodeB, meaning all resources are used for payload data only. The decoder employs the Max Log MAP algorithm and is set to a maximum iteration number of $i_{\max} = 8$. CRC24 is employed for High SNR Early Termination (ET).

The computational effort will be measured as *equivalent iterations* of the conventional decoder until termination. Thus for the conventional decoder this measure equals the average number of full iterations until ET. For the BSD approach, this corresponds to the averaged number of iterations, where each iteration is weighted with the percentage of erroneous blocks. For example, if for the BSD approach the decoder would terminate after 3 full iterations, but each constituent decoder would have only processed 50 % of the code block during each full iteration, then this would correspond to 1.5 equivalent iterations.

There are two factors which have major influence on the performance of the BSD approach:

– *Code rate* The code rate determines how many positions are punctured in the transmitted block. The higher the code rate, the more punctured positions and consequently the more depunctured, zero-reliability bits are input to the decoder. However, the reliability of the syndrome-based criterion for identification of error-free blocks depends on a good estimate of the parity sequence, because of the involved precorrection. Thus, if there are more zero-reliability parity positions this criterion is less reliable and a larger value has to be selected for $\ell_{\min}$ to avoid a loss of BLER.

– *Channel type* The type of the channel influences the convergence behavior of the Turbo decoder. Naturally the BSD approach is more effective if the decoder executes more iterations before termination, because in this case the BSD has more chances to identify error-free parts.

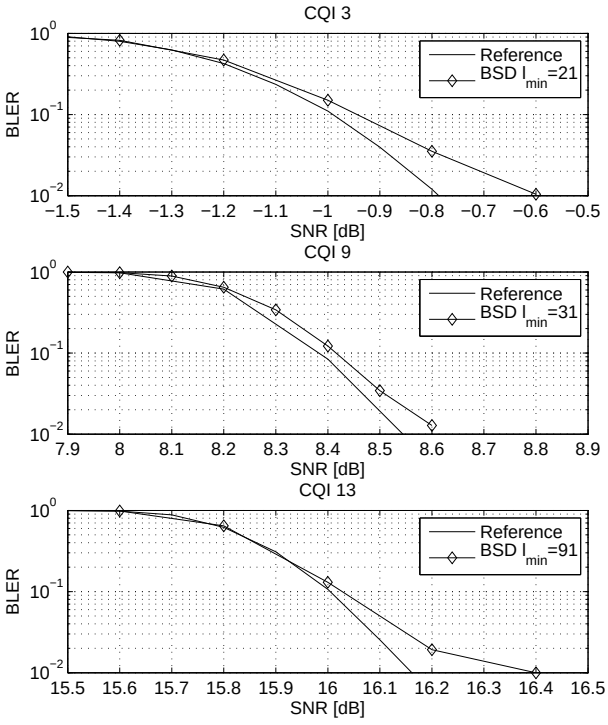


**Fig. 2.** Comparison of BLER for different CQI values (AWGN).

The following Sect. 4.1 compares BLER and equivalent iterations for different CQI values, i.e. different code rates. The influence of the transmission channel is analyzed in Sect. 4.2. A conventional Turbo decoder is used to provide reference results.

### 4.1 Code rates

To illustrate the performance of the BSD approach for different code rates, the CQI values 3, 9 and 13 are used under the assumption of AWGN transmission. The parameter $\ell_{\min}$ is adjusted such that the BLER degradation is smaller than

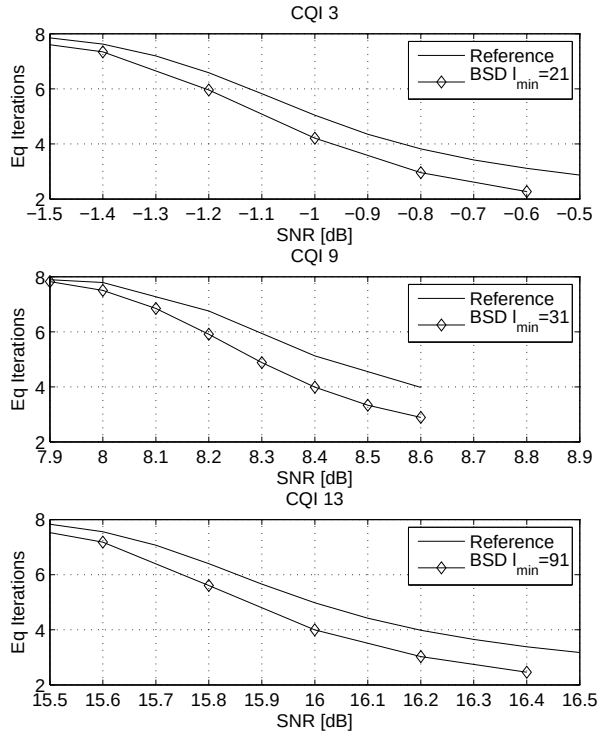


**Fig. 3.** Equivalent iterations for different CQI values (AWGN).

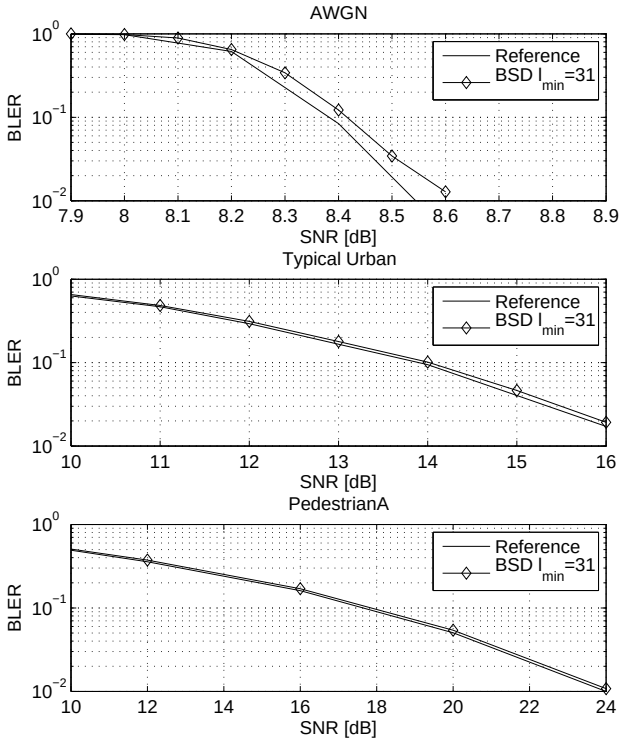


**Fig. 4.** Comparison of BLER for different channels (CQI9).

0.1 dB compared to the reference; it is set to $\ell_{\min} = 21$ for CQI3, $\ell_{\min} = 31$ for CQI9 and $\ell_{\min} = 91$ for CQI13. As already mentioned, a larger value is required for a higher CQI (higher code rate). The resulting BLER for the three cases is shown on Fig. 2. It can be noted, that there is a slight performance decrease, which however is still smaller than the required 0.1dB at a working point of 10 % BLER.

The equivalent number of iterations that are executed by the decoder are shown in Fig. 3. Again, the BSD approach is compared to the conventional Turbo decoder for the considered SNR range. For the conventional decoder the equivalent number of iterations only depends on the CRC24 based ET. For the BSD, it is additionally influenced by the amount of error-free subblocks that are identified during the iterations.

Naturally, the number of iterations that are required to decode a block decreases with increasing SNR for both decoders, because the quality of the received values improves. For the BSD decoder the number of equivalent iterations is always smaller than that of the conventional decoder. For example, considering a working point of 10 % BLER, the computational effort is reduced by about 0.7, 1.1 and 1 equivalent iterations at CQI3, CQI9 and CQI13. This corresponds to a reduction of decoding operations by about 20%.

### 4.2 Channels

In order to analyze the impact of different channel conditions, Figs. 4 and 5 compare the performance of BSD over AWGN channel to the performance over the fading channels Typical Urban (TU) and PedestrianA (PedA). The BSD is shown along with the conventional decoder for a fixed CQI of 9. The BLER is shown in Fig. 4 for the three cases. As in the previous simulation, the parameter $\ell_{\min}$ has been selected such that the impact on BLER performance is $< 0.1dB$ at the BLER working point of 10 %. For all channels the design parameter is set to $\ell_{\min} = 31$.

The reduction of decoding effort in terms of equivalent iterations is shown in Fig. 5. It can be noticed, that for the fading channels TU and PedA the average number of iterations of the conventional Turbo decoder is much smaller than in the AWGN case: While for the AWGN channel the decoder executes about 5.1 iterations in average at the working point of 10% BLER, it only executes about 2.2 and 1.9 iterations for the TU and PedA channel, respectively. This clearly affects the efficiency of the BSD approach, because if less iterations are executed until ET, there are less options for the BSD to identify error-free blocks, and thus less reduction of computational effort. Consequently the reduction in terms of equivalent iterations is about 0.5 and 0.2 for TU channel and PedA channel and thus smaller compared to the AWGN case.

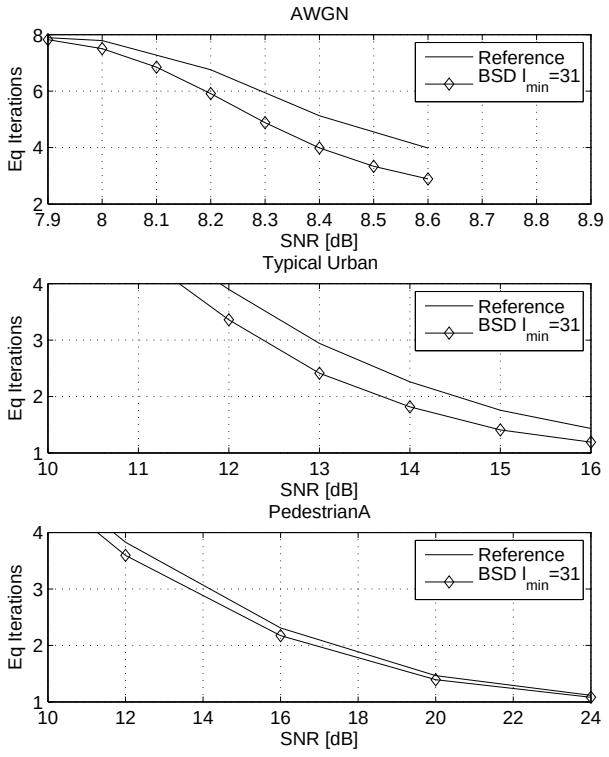


**Fig. 5.** Equivalent iterations for different channels (CQI9).

Table 1 summarizes the results presented in this section. For a BLER working point of 10 % and for the different CQIs and channel types, the equivalent iterations of conventional Turbo decoder and BSD are shown along with the used parameter $\ell_{\min}$. Additionally the last column shows the relative reduction of equivalent iterations. It can be seen, that in case of AWGN and TU channel the reduction of computational effort is about 20%, while in case of the PedA channel it is only 6%. It can further be noticed, that the choice of the parameter $\ell_{\min}$ depends on the CQI, i.e. the code rate, and not on the channel type. Further simulations have also shown that the variation of other system parameters has no significant influence on the choice of $\ell_{\min}$ either. Thus, the setting of $\ell_{\min}$ can be done by the decoder based on the current MCS reporting.

## 5 Conclusions

In this paper a decoding algorithm for Turbo Codes with reduced computational complexity has been described. The approach is based on a syndrome based identification of supposedly error-free blocks in the input sequence of a constituent decoder. Decoding only the remaining erroneous blocks results in a reduction of decoding effort while keeping the impact on the BLER performance insignificant. Depending on the channel type, the computational effort measured in equivalent iterations of the conventional decoder can be reduced by up to 20%. The approach is especially effective if the channel requires a relatively high number of iterations until termination, like e.g. in the AWGN case. On the other hand, if the channel only requires a small number of iterations (like 2–3 iterations for the PedA and TU channels), the reduction in terms of absolute iterations will be smaller. For these cases, a low SNR ET scheme may be considered, which may also be implemented based on the hamming weight of the syndrome sequence **b** (Geldmacher et al., 2011).

**Table 1.** Summarized results for different CQIs and channels.

| | Code Rate | $\ell_{\min}$ | Eq. It. Conv. Dec. | Eq. It. BSD | Saving |
|---|---|---|---|---|---|
| AWGN, CQI3 | 0.19 | 21 | 3.8 | 3.0 | 21 % |
| AWGN, CQI9 | 0.60 | 31 | 5.1 | 3.9 | 22 % |
| TU, CQI9 | 0.60 | 31 | 2.3 | 1.8 | 20 % |
| PedA, CQI9 | 0.60 | 31 | 2.3 | 2.1 | 6 % |
| AWGN, CQI13 | 0.75 | 91 | 5.0 | 3.8 | 24 % |

The realization of a syndrome based Turbo decoder is of the same complexity as a conventional Turbo decoder, because only the underlying trellis is changed. Additional effort is required for syndrome computation, the generation of an estimate of the parity errors and precorrection. These operations are based only on bits, such that they are easy to realize using XOR operations. Additionally, the identification of error-free blocks has to be done. This may be implemented by a simple counter.

## References

Evolved Universal Terrestrial Radio Access (E-UTRA): 3GPP TS 36.213 V8.8.0, 2009.

Berrou, C., Glavieux, A., and Thitimajshima, P.: Near Shannon limit error-correcting coding and decoding: Turbo-codes, in: IEEE ICC, 1064–1070, Geneva, Switzerland, 1993.

Forney Jr., G.: Convolutional codes I: Algebraic structure, IEEE Transactions on Information Theory, 16, 720–738, 1970.

Geldmacher, J., Hueske, K., and Götze, J.: An Adaptive and Complexity Reduced Decoding Algorithm for Convolutional Codes and its Application to Digital Broadcasting Systems, in: International Conference on Ultra Modern Telecommunications (ICUMT2009), St. Petersburg, Russia, doi:10.1109/ICUMT.2009.5345627, 2009.

Geldmacher, J., Hueske, K., Bialas, S., and Götze, J.: Adaptive Low Complexity MAP Decoding for Turbo Equalization, in: 6th International Symposium on Turbo Codes & Iterative Inf. Proc. (ISTC2010), Brest, France, doi:10.1109/ISTC.2010.5613804, 2010.

Geldmacher, J., Hueske, K., Götze, J., and Kosakowski, M.: Hard Decision Based Low SNR Early Termination for LTE Turbo Decoding, in: 8th International Symposium on Wireless Communication Systems (ISWCS2011), Aachen, Germany, 2011.

Hueske, K., Geldmacher, J., and Götze, J.: Syndrome Based Adaptive Complexity Channel Decoding and Turbo Equalization for ATSC DTV, in: 44th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, 2010.

Mehlführer, C., Wrulich, M., Ikuno, J. C., Bosanska, D., and Rupp, M.: Simulating the Long Term Evolution Physical Layer, in: Proceedings of the 17th European Signal Processing Conference (EUSIPCO 2009), Glasgow, Scotland, 2009.

Minowa, T.: A syndrome decoding of high-rate turbo codes, in: IEEE International Symposium on Information Theory, 384–384, doi:10.1109/ISIT.2003.1228400, 2003.

Robertson, P., Villebrun, E., and Hoeher, P.: A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain, IEEE ICC, 1009–1013, doi:10.1109/ICC.1995.524253, Seattle, 1995.

Schalkwijk, J. and Vinck, A.: Syndrome Decoding of Binary Rate-1/2 Convolutional Codes, IEEE Transactions on Communications, 24, 977–985, doi:10.1109/TCOM.1976.1093420, 1976.

Tajima, M., Shibata, K., and Kawasaki, Z.: Modification of Syndrome Trellises for High-Rate Convolutional Codes, Tech. rep., The Institute of Electronics, Information and Communication Engineers (IEICE), 2002.

Yamada, T., Harashima, H., and Miyakawa, H.: A new maximum likelihood decoding of high rate convolutional codes using a trellis, IEICE Transactions on Electronics and Communications in Japan, J66-A, 611–616, doi:10.1002/ecja.4400660703, 1983.