2022 56th Asilomar Conference on Signals, Systems, and Computers | 978-1-6654-5906-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/IEEECONF56349.2022.10051944

DUIDD: Deep-Unfolded Interleaved Detection and Decoding for MIMO Wireless Systems

Reinhard Wiesmayr[1], Chris Dick[2], Jakob Hoydis[2], and Christoph Studer[1]

[1]*ETH Zurich, Switzerland; wiesmayr@iis.ee.ethz.ch and studer@ethz.ch*
[2]*NVIDIA Corporation; cdick@nvidia.com and jhoydis@nvidia.com*

***Abstract*—Iterative detection and decoding (IDD) is known to achieve near-capacity performance in multi-antenna wireless systems. We propose deep-unfolded interleaved detection and decoding (DUIDD), a new paradigm that reduces the complexity of IDD while achieving even lower error rates. DUIDD interleaves the inner stages of the data detector and channel decoder, which expedites convergence and reduces complexity. Furthermore, DUIDD applies deep unfolding to automatically optimize algorithmic hyperparameters, soft-information exchange, message damping, and state forwarding. We demonstrate the efficacy of DUIDD using NVIDIA's Sionna link-level simulator in a 5G-near multi-user MIMO-OFDM wireless system with a novel low-complexity soft-input soft-output data detector, an optimized low-density parity-check decoder, and channel vectors from a commercial ray-tracer. Our results show that DUIDD outperforms classical IDD both in terms of block error rate and computational complexity.**

I. Introduction

With the development of recent and future communication standards, such as 5G [1] and 6G [2], basestations (BSs) are required to process data at higher rates from a larger number of antennas and users while achieving higher spectral efficiency and fulfilling stricter quality-of-service requirements. In order to keep up with these trends, the design of novel low-complexity multiple-input multiple-output (MIMO) data detection and channel decoding algorithms is necessary.

Wireless receivers for massive multi-user (MU) MIMO systems typically resort to linear miminum mean-square error (LMMSE) data detection followed by a separate channel decoding stage [3]–[9]. Albeit practical to implement, such conventional pipelines that separate data detection from channel decoding are known to operate far from the fundamental capacity limits. In contrast, iterative detection and decoding (IDD) is known to achieve near-capacity performance in MIMO wireless systems [10]. However, the complexity of IDD is already high in conventional, small-scale MIMO systems [11]–[13], which likely prevents a deployment in multi-antenna wireless systems that must support more antennas, more simultaneously transmitting users (and possibly more parallel data streams), and higher bandwidths [14].

The work of RW and CS was supported in part by an ETH Research Grant. The work of CS was supported in part by ComSenTer, one of six centers in JUMP, a SRC program sponsored by DARPA. The work of CS was supported in part by the U.S. NSF under grants CNS-1717559 and ECCS-1824379.

The authors thank Gian Marti for discussions and comments on this paper. All code and simulation scripts to reproduce the results of this paper are available on GitHub: https://github.com/IIP-Group/DUIDD

A. Contributions

We propose deep-unfolded interleaved detection and decoding (DUIDD), a novel receiver pipeline for traditional, small scale MIMO as well as massive MU-MIMO wireless systems that surpasses the error rate performance of IDD at lower computational complexity. DUIDD interleaves the inner iterations of the soft-input soft-output (SISO) data detection and channel decoding stages, which expedites convergence, thereby reducing the number of inner and outer iterations. In order to automatically optimize algorithm hyperparameters, soft-information exchange, message damping, and state forwarding, DUIDD combines deep unfolding [15] with emerging machine learning (ML) frameworks [16] and block error rate (BLER)-based training [17]. Besides utilizing the SISO minimum mean-square-error (MMSE) parallel interference cancellation (PIC) data detector [18], [19], we also propose a novel low-complexity SISO data detector that further reduces complexity. For channel decoding, we consider a low-density parity-check (LDPC) decoder with message damping [20], [21]. We demonstrate the effectiveness of DUIDD by comparing the BLER performance and computational complexity of classical IDD and DUIDD in a 5G-near MIMO orthogonal frequency-division multiplexing (OFDM) system simulated with NVIDIA's Sionna [16].

B. Related Work

The importance of artificial intelligence (AI) and ML in wireless communications is growing quickly and already an integral part of 5G Release 18 [22]. AI and ML are expected to play and even more central role in 6G systems [2] and research is heavily focusing on ML-based baseband processing, ranging from atomistic (separate and independent) optimizations of neural network (NN)-based MIMO data detectors [23]–[25] or channel decoders [26], to fully NN-based transmitters and receivers (e.g., end-to-end learning methods [27]–[30]), and model-driven deep unfolding approaches [20], [21], [31]. However, existing NN-based data detectors turn out to require orders of magnitude higher complexity compared to classical algorithms that were designed by hand. For example, even when excluding training, the NN-based data detectors compared in [32, Tbl. I] require $5\times$ to more than $100\times$ higher runtime than a standard LMMSE data detector. In contrast, model-driven methods which utilize deep unfolding and ML for automated parameter tuning enable superior error rate performance without increasing the computational complexity.

For the same reason, deep unfolding was utilized to optimize either classical data detection or channel decoding algorithms by hyperparameter tuning, or augment classical algorithms with a hypernetwork. Both methods were applied in [31] for improving a classical expectation propagation (EP) data detector in a deep-unfolded IDD receiver. Other examples for training a traditional LDPC decoder include message damping and weighted belief-propagation decoding [20], [21]. However, to the best of our knowledge, there exists no prior work that jointly optimizes (iterative) detection and decoding pipelines on all the system and component levels, which is part of what we do with the proposed DUIDD paradigm.

C. Notation

Column vectors are written in bold lowercase, matrices in bold uppercase, and sets in calligraphic letters. The superscript $^{-1}$ denotes the matrix inverse, $^{-\frac{1}{2}}$ the inverse matrix square root, and ^{H} the Hermitian. In general, we use subscripts for indexing (e.g., x_u refers to user equipment u), but denote the K-dimensional all-zero vector by $\mathbf{0}_K$, the $M \times M$ identity matrix is $\mathbf{I}_M$, the $M \times K$ all-ones matrix is $\mathbf{1}_{M\times K}$, and the K-dimensional all-ones vector $\mathbf{1}_K$; we omit the dimensions whenever they are obvious in the context. We refer to the kth entry of the uth vector $\mathbf{x}_u$ by $[\mathbf{x}_u]_k$, to the mth row of a matrix $\mathbf{X}$ by $[\mathbf{X}]_{m,:}$, and to the element in the mth row and nth column by $[\mathbf{X}]_{m,n}$. The diagonal matrix $\text{diag}(\mathbf{x})$ contains the elements of $\mathbf{x}$ on the main diagonal. The operator $\mathbb{P}[\cdot]$ denotes probability, $\mathbb{E}[\cdot]$ expectation, $\mathbb{V}\text{ar}[\cdot]$ variance, and $\mathbb{C}\text{ov}[\cdot]$ covariance. We use $\mathcal{NC}(\mathbf{0},\mathbf{C})$ to refer to circularly-symmetric complex Gaussian distribution with covariance matrix $\mathbf{C}$.

II. System Model

We consider a coded MU-MIMO-OFDM wireless uplink system in which U single-antenna user equipments (UEs) simultaneously transmit data to one B-antenna BS. We now detail the encoder and the system model.

A. OFDM-Based MU Data Transmission

Within an OFDM frame, each UE $u \in \{1,\dots,U\}$ transmits D data bits $\mathbf{d}_u \in \{0,1\}^D$, which are separately (for each UE) encoded to K coded bits following the encoding rule $\mathbf{b}_u = \text{enc}(\mathbf{d}_u) \in \{0,1\}^K$ with code rate $R = \frac{D}{K}$. The encoded bits are mapped onto the zero-mean unit-variance transmit constellation points from the set $\mathcal{Q}$ (e.g., QPSK), to data vectors $\mathbf{s}_u \in \mathcal{Q}^V$, where each symbol carries $Q = \log_2|\mathcal{Q}|$ bits and $V = \frac{K}{Q}$. The V symbols are mapped onto the OFDM resource grid with W data-carrying sub-carriers and T OFDM symbols, which are divided into T_D data symbols and $T_P = T - T_D$ pilots (used for channel estimation); we have $V = WT_D$. Thus, within each OFDM frame (comprising of WT resource elements), all UEs simultaneously transmit one codeword $\mathbf{b}_u$ and pilots for channel estimation over the wireless channel.

B. MU-MIMO-OFDM System Model

For OFDM resource element (RE) $v \in \{1,\dots,WT\}$, we model the frequency-flat baseband input-output relation as

$$\mathbf{y}_v = \mathbf{H}_v\mathbf{s}_v + \mathbf{n}_v, \qquad (1)$$

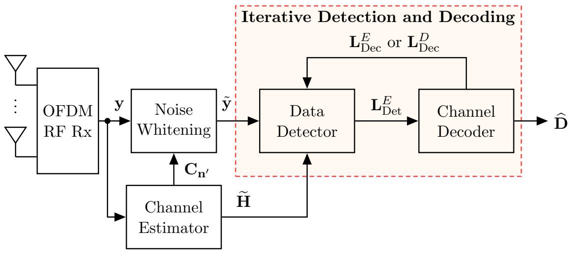


Fig. 1. Classical iterative detection and decoding (IDD) MIMO receiver.

where $\mathbf{y}_v \in \mathbb{C}^B$ is the BS-side receive vector, $\mathbf{H}_v \in \mathbb{C}^{B\times U}$ is the channel matrix, $\mathbf{s}_v \in \mathcal{Q}^U$ the vector containing the UEs' transmit symbols within RE v, with $[\mathbf{s}_v]_u$ corresponding to the symbol of UE u, and $\mathbf{n}_v \in \mathbb{C}^B$ models thermal noise with distribution $\mathcal{NC}(\mathbf{0}, \text{N}_0\mathbf{I})$. In what follows, we omit the RE index v to simplify notation.

Since practical systems must estimate the channel matrix $\mathbf{H}$, we model the *estimated* channel matrix as $\widehat{\mathbf{H}} = \mathbf{H} - \mathbf{E}$ with estimation error matrix $\mathbf{E}$. This model enables us to rewrite the input-output relation in (1) as follows:

$$\mathbf{y} = \left(\widehat{\mathbf{H}} + \mathbf{E}\right)\mathbf{s} + \mathbf{n} = \widehat{\mathbf{H}}\mathbf{s} + \mathbf{n}'. \qquad (2)$$

Here, $\mathbf{n}' = \mathbf{E}\mathbf{s} + \mathbf{n}$ is the effective noise vector with covariance

$$\mathbf{C}_{\mathbf{n}'} = \mathbb{C}\text{ov}[\mathbf{n}'] = \mathbb{C}\text{ov}[\mathbf{E}\mathbf{s}+\mathbf{n}] = \mathbf{C}_{\mathbf{E}} + \text{N}_0\mathbf{I} \qquad (3)$$

and $\mathbf{C}_{\mathbf{E}} = \mathbb{C}\text{ov}[\mathbf{E}\mathbf{s}]$ is the channel estimation error covariance, which is assumed to be independent of the realization of $\mathbf{s}$ and provided by the channel estimator. Since the effective noise vector $\mathbf{n}'$ is correlated, we design our MIMO data detectors for the whitened system model

$$\tilde{\mathbf{y}} = \mathbf{C}_{\mathbf{n}'}^{-\frac{1}{2}}\mathbf{y} = \mathbf{C}_{\mathbf{n}'}^{-\frac{1}{2}}\left(\widehat{\mathbf{H}}\mathbf{s} + \mathbf{n}'\right) = \widetilde{\mathbf{H}}\mathbf{s} + \tilde{\mathbf{n}}, \qquad (4)$$

in which $\widetilde{\mathbf{H}} = \mathbf{C}^{-\frac{1}{2}}\widehat{\mathbf{H}}$ is the whitened channel estimate and $\tilde{\mathbf{n}} = \mathbf{C}_{\mathbf{n}'}^{-\frac{1}{2}}\mathbf{n}' \sim \mathcal{NC}(\mathbf{0},\mathbf{I})$ the whitened noise vector.

III. Iterative Detection and Decoding

We now summarize a conventional IDD receiver for MIMO wireless systems, which will serve as a starting point for the proposed DUIDD paradigm introduced in Sec. IV.

A. Classical IDD Receiver

After receiving an OFDM frame, the BS first performs channel estimation and noise whitening; see Fig. 1 for an illustration. Then, the BS provides the whitened receive vector $\tilde{\mathbf{y}}$ and channel estimate $\widetilde{\mathbf{H}}$ to a MIMO data detector, which calculates (approximate) extrinsic log-likelihood ratios (LLR) values $\mathbf{L}^E_{\text{Det}} \in \mathbb{R}^{U\times K}$ (also known as logits) for the U UEs and K coded bits. A SISO channel decoder then utilizes these LLR values to calculate (hopefully improved) extrinsic LLR values $\mathbf{L}^E_{\text{Dec}} = \text{dec}(\mathbf{L}^E_{\text{Det}}) \in \mathbb{R}^{U\times K}$ for every coded bit[1].

In a classical IDD receiver (see Fig. 1), the extrinsic LLR values are then fed back to the data detector, where they

[1]Since the UEs independently encode their bits $\mathbf{d}_u$, the decoder *separately* decodes $[\mathbf{L}^E_{\text{Dec}}]_{u,:} = \text{dec}([\mathbf{L}^E_{\text{Det}}]_{u,:}) \in \mathbb{R}^K$ in the considered MU scenario.

serve as a-priori LLR values, i.e., $\mathbf{L}_{\text{Det}}^{A} = \mathbf{L}_{\text{Dec}}^{E}$ [10]. A SISO data detector then uses $\tilde{\mathbf{y}}$, $\widetilde{\mathbf{H}}$, and $\mathbf{L}_{\text{Det}}^{A}$ to compute new (and hopefully improved) extrinsic LLR values that are passed to the SISO channel decoder. The channel decoder produces new LLR values and this process is repeated I times, where $I = 1$ means that we use the data detector and channel decoder once.

As outlined above, the data detector takes in a-priori LLR values from the channel decoder, which we define as

$$[\mathbf{L}_{\text{Det}}^{A}]_{u,k} = \log\left(\frac{\mathbb{P}[b_{u,k} = 1]}{\mathbb{P}[b_{u,k} = 0]}\right) \tag{5}$$

for code bit k of UE u. The data detector then utilizes the whitened receive vector and channel matrix to compute soft-outputs. By considering a-priori LLR information, the SISO data detector typically first computes intrinsic LLR values that model the a-posteriori bit probabilities, i.e.,

$$[\mathbf{L}_{\text{Det}}^{D}]_{u,k} = \log\left(\frac{\mathbb{P}[b_{u,k} = 1|\tilde{\mathbf{y}}, \widetilde{\mathbf{H}}, \mathbf{L}_{\text{Det}}^{A}]}{\mathbb{P}[b_{u,k} = 0|\tilde{\mathbf{y}}, \widetilde{\mathbf{H}}, \mathbf{L}_{\text{Det}}^{A}]}\right). \tag{6}$$

While the intrinsic LLR values are most informative about the current bit-probability belief, the *extrinsic* LLRs model the information gain from the detector, which are usually forwarded to the decoder. By applying Bayes' rule to (6), the data detector passes *extrinsic* LLR values to the channel decoder as [10]

$$\mathbf{L}_{\text{Dec}}^{E} = \mathbf{L}_{\text{Dec}}^{D} - \mathbf{L}_{\text{Dec}}^{A}. \tag{7}$$

While exchanging extrinsic LLR values is the de-facto standard, some data detection algorithms (e.g., the SISO MMSE-PIC algorithm) perform better when *intrinsic* LLR values are fed back from the channel decoder to the data detector [33], i.e., one directly uses $\mathbf{L}_{\text{Det}}^{A} = \mathbf{L}_{\text{Dec}}^{D}$. This subtle yet important aspect will be discussed later in Sec. IV-B.

After I iterations, the channel decoder for UE u produces hard-output estimates for the transmitted data bits according to $\hat{\mathbf{d}}_u^T = [\hat{\mathbf{D}}]_{u,:} = \text{u}(\text{dec}^*([\mathbf{L}_{\text{Det}}^{E}]_{u,:})) \in \{0,1\}^{1\times D}$, where $\text{dec}^*(\cdot)$ is the soft-output decoding rule for the data bits and u slices the LLR values to bits in $\{0,1\}$ according to their signs.

B. SISO MMSE-PIC Algorithm

The SISO MMSE-PIC algorithm is a practical SISO data detector for classical IDD receivers. This algorithm achieves near-capacity performance at manageable complexity and was first described in [18] for systems with inter-symbol interference. Reference [19] reduced the algorithm's complexity without altering the result, which is the method we detail next. For each data-carrying OFDM RE $v \in \{1,\dots,V\}$, the SISO MMSE-PIC algorithm carries out the following four steps where we omit the RE index v for simplicity.

1) Soft-Symbol Computation: The algorithm first converts the a-priori LLR values pertaining to UE u in $\mathbf{L}_{\text{Det}}^{A}$ into a-priori soft-symbol estimates $\hat{s}_u^A$ and corresponding variances $\hat{\nu}_u^{2,A}$. We define the soft symbol as $\hat{s}_u^A = s_u - e_u^A$, i.e., the true transmit symbol s_u perturbed by an error e_u^A, which is assumed to be zero mean with variance $\hat{\nu}_u^{2,A}$. We compute the soft symbol and variance as follows:

$$\hat{s}_u^A = \textstyle\sum_{a\in\mathcal{Q}} a\,\mathbb{P}[s_u = a|\mathbf{L}_{\text{Det}}^{A}] \tag{8}$$

$$\hat{\nu}_u^{2,A} = \textstyle\sum_{a\in\mathcal{Q}} \left|a - \hat{s}_u^A\right|^2 \mathbb{P}[s_u = a|\mathbf{L}_{\text{Det}}^{A}], \tag{9}$$

with $\mathbb{P}[s_u = a|\mathbf{L}_{\text{Det}}^{A}]$ denoting the probability that s_u is equal to constellation symbol a, which we compute by

$$\mathbb{P}[s_u = a|\mathbf{L}_{\text{Det}}^{A}] = \textstyle\prod_{q=1}^{Q} \mathbb{P}[b_{u,q} = k_{a,q}|[\mathbf{L}_{\text{Det}}^{A}]_{u,q}]. \tag{10}$$

Here, $k_{a,q}$ denotes the qth bit of constellation symbol a. We compute the right-hand-side probability as

$$\mathbb{P}[b_{u,q} = k_{a,q}|[\mathbf{L}_{\text{Det}}^{A}]_{u,q}] = \sigma\big((2k_{a,q} - 1)[\mathbf{L}_{\text{Det}}^{A}]_{u,q}\big) \tag{11}$$

with the logistic sigmoid $\sigma(x) = 1/(1+e^{-x})$.

2) Parallel Interference Cancellation (PIC): Starting from the whitened system model in (4), the algorithm now performs PIC using the computed soft symbols and error terms for UE u by subtracting the interference of all other users

$$\tilde{\mathbf{y}}_u = \tilde{\mathbf{y}} - \sum_{u'\neq u}^{U} \tilde{\mathbf{h}}_{u'}\hat{s}_{u'}^A = \tilde{\mathbf{h}}_u s_u + \sum_{u'\neq u}^{U} \tilde{\mathbf{h}}_{u'} e_{u'}^A + \tilde{\mathbf{n}}. \tag{12}$$

3) MMSE Equalization: Considering the model after PIC in (12), we now perform LMMSE equalization to estimate s_u. We follow the approach in [19] and compute

$$\mathbf{A}^{-1} = \big(\widetilde{\mathbf{H}}^H\widetilde{\mathbf{H}}\mathbf{\Lambda} + \mathbf{I}\big)^{-1}, \tag{13}$$

with $\mathbf{\Lambda} = \text{diag}\big(\hat{\nu}_1^{2,A},\dots,\hat{\nu}_U^{2,A}\big)$. The LMMSE equalization matrix is given by $\mathbf{W} = \mathbf{A}^{-1}\widetilde{\mathbf{H}}^H$, which we use to calculate an *unbiased* estimate for s_u as follows:

$$\hat{s}_u^D = \tfrac{1}{\mu_u}\mathbf{w}_u^H\tilde{\mathbf{y}}_u. \tag{14}$$

Here, $\mathbf{w}_u^H$ is the uth row of $\mathbf{W}$ and $\mu_u = \mathbf{w}_u^H\tilde{\mathbf{h}}_u$ is the bias. As shown in [19, App. C], the post-equalization noise-plus-interference (NPI) variance of $\hat{s}_u^D$ can be computed as

$$\hat{\nu}_u^{2,D} = \tfrac{1}{\mu_u} - \hat{\nu}_u^{2,A}. \tag{15}$$

4) LLR Calculation: We now use the symbol estimate in (14) to calculate extrinsic max-log LLR values as [19, Sec. III-B]

$$[\mathbf{L}_{\text{Det}}^{E}]_{u,q} = \frac{1}{\hat{\nu}_u^{2,D}}\Big(\min_{a\in\mathcal{Q}_q^0}\left|\hat{s}_u^D - a\right|^2 - \min_{a\in\mathcal{Q}_q^1}\left|\hat{s}_u^D - a\right|^2\Big), \tag{16}$$

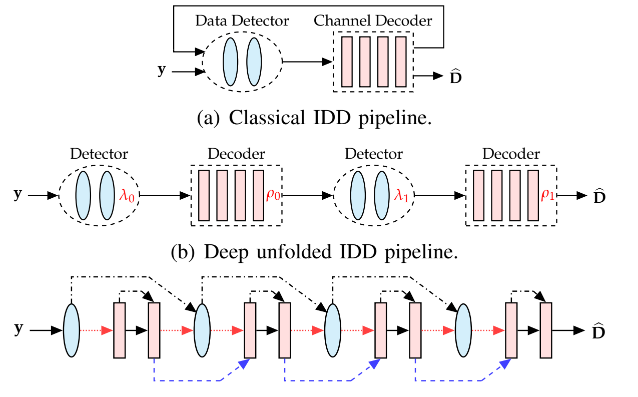
where $\mathcal{Q}_q^b$ denotes the set of constellation symbols in which the qth bit is equal to b.

C. LDPC Message Passing Decoding

Throughout this paper, we focus on 5G LDPC forward error-correcting codes [1] with message passing (MP) decoding. We note that IDD and DUIDD can, in principle, be used together with any other SISO channel decoder. The SISO LDPC decoder takes in extrinsic LLR values $\mathbf{L}_{\text{Det}}^{E}$ from the data detector, performs MP for N_i inner iterations, where $i = 1,\dots,I$ is the outer IDD iteration index, and computes posterior LLR values $\mathbf{L}_{\text{Dec}}^{D}$. For all I IDD iterations, the number of total MP iterations is $N_{\text{MP}} = \sum_{i=1}^{I} N_i$. More details about the decoding procedure and our modifications are provided in Sec. IV-D.

IV. Deep Unfolding and Interleaving of IDD

We now introduce the ideas behind DUIDD and then detail methods that further improve the efficacy of this paradigm.



(a) Classical IDD pipeline.

(b) Deep unfolded IDD pipeline.

(c) Deep unfolded interleaved detection and decoding (DUIDD).

Fig. 2. Derivation of DUIDD from classical IDD.

A. *DUIDD: Ideas and Concept*

DUIDD breaks the strict separation between data detector and channel decoder in a traditional IDD receiver and jointly optimizes all hyperparameters using deep unfolding and ML to minimize the system's BLER. The key ideas of DUIDD are illustrated in Fig. 2 and detailed next.

1) Deep-unfolding of IDD: Fig. 2(a) shows a classical IDD receiver pipeline with strict separation between data detector and channel decoder. Note that both components may contain inner iterations (shown with blue ellipses and red rectangles), e.g., the MP iterations of a LDPC decoder. As shown in Fig. 2(b), DUIDD first applies deep unfolding [15] to the IDD receiver for a fixed number of outer iterations I. All hyperparameters per inner iteration $i = 1, \ldots, I$ of the channel decoder and data detector λ_i and ρ_i, respectively, can then be optimized automatically using ML tools.

2) DUIDD: As illustrated in Fig. 2(c), the key idea is to break the strict separation of the deep unfolded IDD receiver in Fig. 2(b) and interleave (reshuffle) the inner iterations of the data detector and channel decoder. The reasons behind interleaving are as follows. First, interleaving enables an earlier information exchange between the detector and decoder, which can accelerate convergence. Convergence can be optimized by selecting a suitable interleaving pattern, i.e., by specifically deciding on the order between inner detection and decoding stages. Second, one can optimize the information exchange across the entire DUIDD pipeline, e.g., by passing extrinsic or intrinsic LLR values (or a mix thereof; see the red dotted arrows), state forwarding (blue dashed arrows), and message damping (black dash-dotted arrows). In the ensuing discussion, we apply the DUIDD paradigm to the IDD receiver reviewed in Sec. III and derive the structure, components, and trainable hyperparameters (highlighted with red) detailed in Fig. 3.

B. *Exchange of Soft-Information*

In classical IDD, the data detector and channel decoder blocks typically exchange extrinsic LLR values. In practice, however, some data detectors are known to perform better when fed with intrinsic a-priori LLR values, such as the SISO MMSE-PIC algorithm [33]. Furthermore, when using approximate detector or decoder blocks (e.g., max-log turbo decoders), it is known that scaling the extrinsic LLR values can improve convergence of iterative methods [34]. We therefore propose to automatically learn the type and scaling of information exchange by computing the a-priori LLR values for the data detector and channel decoder as follows:

$$\mathbf{L}^{A}_{\text{Det}} = \alpha_i \mathbf{L}^{D}_{\text{Dec}} - \beta_i \mathbf{L}^{A}_{\text{Dec}} \tag{17}$$

$$\mathbf{L}^{A}_{\text{Dec}} = \delta_i \mathbf{L}^{E}_{\text{Det}} - \varepsilon_i \mathbf{L}^{A}_{\text{Det}}. \tag{18}$$

Here, we introduce new per-outer-iteration hyperparameters $\{\alpha_i, \beta_i, \delta_i, \varepsilon_i\}$, $i = 1, \ldots, I$. We note that a conventional extrinsic information exchange is realized by setting $\alpha_i = \beta_i = \delta_i = 1$ and $\varepsilon_i = 0$, whereas the MMSE-PIC algorithm performs better with $\alpha_i = \delta_i = 1$ and $\beta_i = \varepsilon_i = 0$. We will use the latter parameters when comparing to classical IDD and to initialize hyperparameter training for DUIDD.

C. *Low-Complexity Soft-Input Soft-Output MIMO Detection*

In a DUIDD receiver, it is beneficial to reduce the complexity of the inner detection and decoding stages as finer granularity can improve the efficacy of interleaving. We now review known methods that reduce complexity of the SISO MMSE-PIC algorithm and then devise a novel, low-complexity variant.

1) Low-Complexity SISO MMSE-PIC: For the Gray-labeling specified in 5G [1], we combine the simplified soft-symbol and variance computation approach from [35] with the approach in [36, Tbl. 1] to directly compute extrinsic max-log LLR values at low complexity. Analogous to [19, Alg. 1], we further reduce complexity by rewriting the SISO MMSE-PIC algorithm to operate on the matched filter (MF) output $\tilde{\mathbf{y}}^{\text{MF}} = \widetilde{\mathbf{H}}^H \tilde{\mathbf{y}}$.

2) SISO LoCo-PIC Algorithm: Since the computational complexity of the SISO MMSE-PIC algorithm is dominated by computing $\mathbf{A}^{-1}$ in (13) whenever the a-priori LLR values change, it can be advantageous to derive a low-complexity variant. We propose a low-complexity (LoCo) version of the SISO MMSE-PIC algorithm based on the observation that there exist two corner cases: (i) no useful a-priori information (i.e., all LLR values are zero) and (ii) perfect a-priori information (i.e., the LLR values determine the ground-truth coded bits).

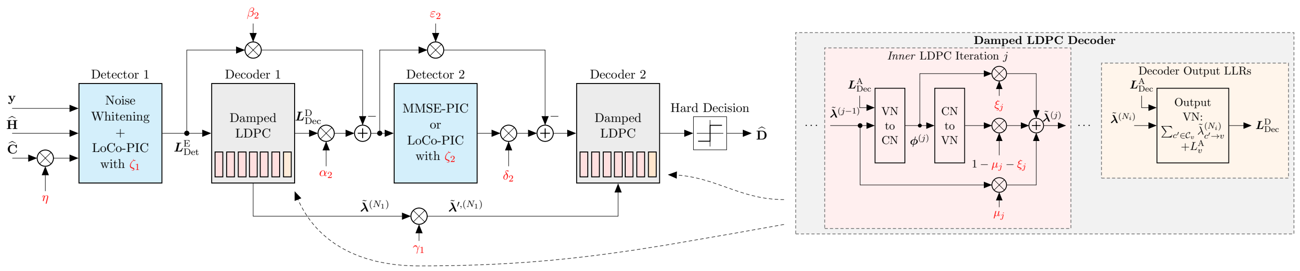
In absence of useful a-priori information, which is generally the case in the first outer iteration, we have $\mathbf{L}^{A}_{\text{Det}} = \mathbf{0}$ with the resulting soft-symbol estimates $\hat{\mathbf{s}}^{A} = \mathbf{0}$ and error variances $\hat{\boldsymbol{\nu}}^{2,A} = \mathbf{1}$. In this case, we obtain the conventional LMMSE filter matrix given by [3]

$$\mathbf{W}_{\text{LM}} = \left(\widetilde{\mathbf{H}}^H \widetilde{\mathbf{H}} + \mathbf{I}\right)^{-1} \widetilde{\mathbf{H}}^H. \tag{19}$$

With perfect a-priori information, the soft-symbol estimates correspond to the true transmit symbols and the error variances are zero. In this case, we obtain the MF matrix $\mathbf{W}_{\text{MF}} = \widetilde{\mathbf{H}}^H$. The idea of LoCo-PIC is now to model the filter matrix as a linear interpolation between these two corner cases:

$$\mathbf{W}_{\text{LC}} = \zeta \mathbf{W}_{\text{LM}} + (1 - \zeta) \mathbf{W}_{\text{MF}}. \tag{20}$$

Here, the parameter ζ indicates whether we use more or less of the LMMSE filter and MF matrices, respectively. While one



β_2
ε_2
Detector 1
Decoder 1
Detector 2
Decoder 2
$\mathbf{y}$
$\widehat{\mathbf{H}}$
$\widehat{\mathbf{C}}$
η
Noise Whitening + LoCo-PIC with ζ_1
$\boldsymbol{L}_{\text{Det}}^{\text{E}}$
Damped LDPC
$\boldsymbol{L}_{\text{Dec}}^{\text{D}}$
α_2
MMSE-PIC or LoCo-PIC with ζ_2
δ_2
Damped LDPC
Hard Decision
$\widehat{\mathbf{D}}$
$\tilde{\boldsymbol{\lambda}}^{(N_1)}$
$\tilde{\boldsymbol{\lambda}}'^{,(N_1)}$
γ_1
Damped LDPC Decoder
Inner LDPC Iteration j
$\boldsymbol{L}_{\text{Dec}}^{\text{A}}$
$\tilde{\boldsymbol{\lambda}}^{(j-1)}$
VN to CN
$\boldsymbol{\phi}^{(j)}$
CN to VN
ξ_j
$1-\mu_j-\xi_j$
μ_j
$\tilde{\boldsymbol{\lambda}}^{(j)}$
Decoder Output LLRs
$\boldsymbol{L}_{\text{Dec}}^{\text{A}}$
$\tilde{\boldsymbol{\lambda}}^{(N_i)}$
Output VN: $\sum_{c'\in\mathcal{C}_v}\tilde{\lambda}_{c'\to v}^{(N_i)} + L_v^{\text{A}}$
$\boldsymbol{L}_{\text{Dec}}^{\text{D}}$

Fig. 3. DUIDD receiver pipeline for $I=2$ unfolded IDD iterations each with $N_i=6$ inner LDPC MP decoding iterations.

can derive a closed-form expression for the optimal parameter that minimizes the post-equalization variance when using this filter matrix, evaluating the resulting expression requires high computational complexity. We therefore resort to ML to automatically learn this parameter for every detection stage.

In order to ensure that we perform unbiased equalization when using (20), we perform equalization as follows:

$$\hat{s}^D_{\text{LC},u} = \left(\tfrac{\zeta}{\mu_{\text{LM},u}}\mathbf{w}_{\text{LM},u} + \tfrac{1-\zeta}{\mu_{\text{MF},u}}\mathbf{w}_{\text{MF},u}\right)^H \tilde{\mathbf{y}}_u, \tag{21}$$

with bias $\mu_{\text{LM},u} = \mathbf{w}^H_{\text{LM},u}\tilde{\mathbf{h}}_u$ and $\mu_{\text{MF},u} = \|\tilde{\mathbf{h}}_u\|^2$ and $\mathbf{w}^H_{\text{LM},u}$, $\mathbf{w}^H_{\text{MF},u}$ denote the uth row of $\mathbf{W}_{\text{LM}}$, $\mathbf{W}_{\text{MF}}$, respectively.

Since evaluating the closed-form expression for the post-equalization variance of $\hat{s}^D_{\text{LC},u}$ requires high complexity, we take—as a heuristic—the variance of the matched filter equalizer output $\hat{s}^D_{\text{MF},u}$ obtained by setting $\zeta=0$ in (21) for the LLR calculation. In hindsight that the optimal ζ is expected to be close to 0 for reliable a-priori LLRs, we argue that the NPI variance of $\hat{s}^D_{\text{MF},u}$ is a good approximation for the variance of $\hat{s}^D_{\text{LC},u}$. Considering the error model after PIC in (12), we compute the MF NPI variance as

$$\hat{\nu}^{2,D}_{\text{MF},u} = \sum_{u'\neq u}^{U}\left|\frac{[\widetilde{\mathbf{G}}]_{u,u'}}{[\widetilde{\mathbf{G}}]_{u,u}}\right|^2 \hat{\nu}^2_{u'} + \frac{1}{[\widetilde{\mathbf{G}}]_{u,u}}, \tag{22}$$

where $\widetilde{\mathbf{G}}$ denotes the Gram matrix $\widetilde{\mathbf{G}} = \widetilde{\mathbf{H}}^H\widetilde{\mathbf{H}}$.

Note that when we first use LoCo-PIC, which is in absence of any useful a-priori information, we apply the error variance of the conventional LMMSE filter $\nu^{2,D}_{\text{LM},u} = 1/\mu_{\text{LM},u} - 1$, as the optimal parameter ζ_1 is expected to be close to 1.

We conclude by noting that LoCo-PIC has significantly lower complexity than the conventional MMSE-PIC algorithm, as it reuses the same LMMSE and MF filter matrices for all iterations and during the entire coherence interval.

D. Damped LDPC Message Passing Decoder

We utilize a SISO LDPC decoder that uses the flooding schedule [37]. To improve performance, we utilize message damping for the check node (CN) to variable node (VN) messages $\lambda_{c\to v}$ from CN c to VN v. Since message damping was originally proposed to overcome issues with trapping sets and loopy factor graphs [20], [38], which become problematic after some MP iterations, we modify the damping approaches from [20], [21] as follows. Instead of applying one single *global* damping parameter for all MP iterations, we propose *individual* message damping. We train individual damping parameters $\mu_j \in [0,1]$ for each MP iteration $j \in \{1,\dots,N_{\text{MP}}\}$. Additionally, we also include the CN update to message damping. As shown in the right part of Fig. 3, we implement the update rule

$$\tilde{\lambda}^{(j)}_{c\to v} = (1-\mu_j-\xi_j)\lambda^{(j)}_{c\to v} + \mu_j\tilde{\lambda}^{(j-1)}_{c\to v} + \xi_j\phi^{(j)}_{v\to c} \tag{23}$$

with $\xi_j \in [0,1]$ and the VN to CN message $\phi^{(j)}_{v\to c}$, which we compute as follows:

$$\phi^{(j)}_{v\to c} = L^A_v + \textstyle\sum_{c'\in\mathcal{C}_v\setminus c}\tilde{\lambda}^{(i-1)}_{c'\to v}. \tag{24}$$

Here, $\mathcal{C}_v$ specifies the set of check nodes connected to variable node c, and L^A_v the a-priori LLR for VN v.

The damped MP decoder outputs its intrinsic LLRs by marginalizing the damped CN to VN messages as

$$L^D_v = L^A_v + \textstyle\sum_{c'\in\mathcal{C}_v}\tilde{\lambda}^{(j)}_{c'\to v}, \tag{25}$$

which are forwarded to the detector after N_i MP iterations.

Another important, yet often neglected, aspect of channel decoders with internal states within IDD is whether or not to forward the decoder's state from one IDD iteration to another. For the considered LDPC MP decoder, the CN to VN messages represent the state after each MP iteration j, which we denote by $\tilde{\boldsymbol{\lambda}}^{(j)}$. As detailed in [13, Sec. III-A], a non-resetting LDPC MP decoder (initialized with the previous state $\tilde{\boldsymbol{\lambda}}^{(N_{i-1})}$) performs substantially better than a resetting decoder (initialized with $\mathbf{0}$), particularly for a low number of MP iterations N_i. In contrast, a resetting decoder can result in better error rate performance when N_i is large. Aiming at optimizing the initialization method, we train hyperparameters γ_i which scale the forwarded decoder state, i.e., we initialize decoder $i+1$ with $\tilde{\boldsymbol{\lambda}}'^{,(N_i)} = \gamma_i\tilde{\boldsymbol{\lambda}}^{(N_i)}$. For the classical IDD benchmarks, as well as for the initial value for training, we apply $\gamma_i = 1$.

E. Block Error Rate (BLER) Training

In order to tune all hyperparameters, we aim at minimizing the BLER by using the recent approach from [17]. We modify the last decoder stage to output LLR values for the data bits and convert these into probabilities similar to (11). We then train the hyperparameters in two stages. First, we minimize the empirical mean of the binary cross entropy (BCE) loss

$$l_{\text{BCE}}(d,\hat{p}) = -d\log(\hat{p}) - (1-d)\log(1-\hat{p}), \tag{26}$$

with probability estimate $\hat{p}$ for the data bit d, by averaging over all transmitted bits within a batch (comprising of M OFDM frame transmissions). We then continue with BLER refinement training by using the normalized LogSumExp loss function [17]

$$l_{\text{LSE}}(\mathbf{d}, \hat{\mathbf{p}}) = \log\left(\sum_{k=1}^{D} \exp(l_{\text{BCE}}([\mathbf{d}]_k, [\hat{\mathbf{p}}]_k)) - D + 1\right), \quad (27)$$

with probability estimates $\hat{\mathbf{p}}$ for data bits $\mathbf{d}$, and we average over all transmitted data blocks within a batch. For each batch sample, the signal-to-noise-ratio (SNR) is drawn uniformly at random from the considered SNR training-range.

V. 5G Link-level Simulation in NVIDIA Sionna

We now demonstrate the efficacy of DUIDD in NVIDIA's Sionna 5G link-level simulator [16]. We next discuss the simulation settings and then results for two scenarios: (i) Rayleigh fading channels with perfect channel state information (CSI) and (ii) channels from a ray-tracer with estimated CSI.

A. Simulation and Training Parameters

We simulate a 5G-near MU-MIMO-OFDM uplink system with 100 MHz bandwidth operating at a carrier frequency of 3.75 GHz. We consider $U = 4$ single-antenna UEs and the following two BS-antenna configurations: (i) $B = 4$ BS antennas when simulating Rayleigh-fading channels and (ii) $B = 8$ antennas when using the ray-tracing channels. We use Sionna's $R = 0.5$ rate 5G LDPC code with rate matching and bit-interleaving as described in [1, Sec 5.4.2.2]. Note that we modified the LDPC decoder to include the changes discussed in Sec. IV-D. For each Monte-Carlo trial, we randomly generate $D = 1'200$ data bits and map the coded bits to a 16-QAM constellation with Gray labeling. The OFDM resource grid consists of 60 subcarriers (5 resource blocks, each with 12 sub-carriers), 30 kHz subcarrier spacing, and $T = 14$ OFDM symbols; $T_P = 4$ OFDM symbols are used for pilot-based channel estimation. We use Sionna's Kronecker pilot pattern twice: time slots $t = \{3, 12\}$ for UE 1 and 2, and time slots $t = \{4, 13\}$ for UE 3 and 4. We use least-squares channel estimation with linear interpolation between subcarriers and time slots. We assume perfect power control so that the channel vectors are normalized to have unit average energy per RE.

For BCE-based pre-training and BLER training, we use the Adam optimizer with $2'500$ batches, each consisting of $M = 40$ samples. The initial hyperparameter values correspond to those of classical IDD. The E_b/N_0 range during training is $[-5, 5]$ dB for Rayleigh fading channels and $[-5, 15]$ dB for ray-tracing channels. After training, we evaluate the BLER with 10^5 unseen samples for each SNR value. Note that we train a single set of hyperparameter values, which we fix after training and use for the entire SNR range during testing.

B. Rayleigh Fading Channels and Perfect CSI

Fig. 4(a) shows simulation results for frequency-flat Rayleigh block fading channel matrices, which are independent and identically distributed for each OFDM block. Note that we assume perfect CSI. We use a total number of $N_{\text{MP}} = 12$ LDPC MP decoding iterations, which we evenly interleave with $I \in \{1, \ldots, 4\}$ detector stages. Remember that the total

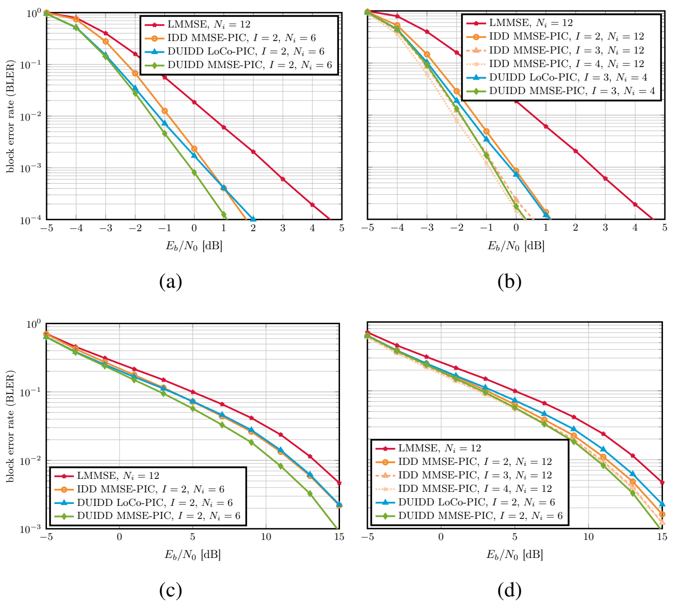


Fig. 4. BLER vs. SNR in a MU-MIMO-OFDM system with $B \times U = 8 \times 4$ Rayleigh fading channels and perfect CSI (top row), or with $B \times U = 16 \times 4$ ray-tracing channels and estimated CSI (bottom row). The left column shows results for a fixed total number of $N_{\text{MP}} = 12$ LDPC MP decoding iterations ($N_i = \frac{12}{I}$ per detection stage). The right column compares DUIDD with $N_i = 12/I$ LDPC MP decoding iterations per detection stage against IDD with a constant ($N_i = 12$) number of iterations per detection stage.

number of MP iterations equals the number of LDPC decoding iterations after each detection stage, i.e., $N_{\text{MP}} = \sum_{i=1}^{I} N_i$. In classical IDD, this situation would correspond to I IDD iterations, each of which applies $N_i = N_{\text{MP}}/I$ decoding iterations. The solid red curve in Fig. 4(a) corresponds to a non-IDD ($I = 1$) baseline with LMMSE detection followed by 12 LDPC decoding iterations. The solid orange curve corresponds to IDD with $I = 2$. The solid blue and solid green lines correspond to DUIDD with LoCo-PIC and MMSE-PIC, respectively, both using $I = 2$. We see that DUIDD with MMSE-PIC performs 0.6 dB better than classical IDD and 2 dB better than the non-iterative receiver at 1% BLER. DUIDD with LoCo-PIC loses only 0.2 dB at 1% BLER.

In Fig. 4(b), we analyze the detection and decoding convergence speed benefits from DUIDD. Thereby, we compare IDD with a constant number of LDPC MP decoding iterations after each detection stage ($N_i = 12$), with $I = 3$ DUIDD with a total number of MP iterations that remains *constant* ($N_{\text{MP}} = 12$, i.e., $N_i = 4$). We see that DUIDD with LoCo-PIC and fewer LDPC decoding iterations achieves approximately the same performance as conventional IDD with MMSE-PIC running for $\{I = 2, N_i = 12\}$ iterations. Even more surprisingly, $\{I = 3, N_i = 4\}$ DUIDD with MMSE-PIC achieves the same performance as IDD with $\{I = 3, N_i = 12\}$. This highlights that DUIDD can significantly increase convergence and thereby decreases the computational complexity.

C. Ray-Tracing Channels and Estimated CSI

Fig. 4(c) shows simulation results for MIMO channel vectors generated using the REMCOM Wireless InSite ray-tracer in the

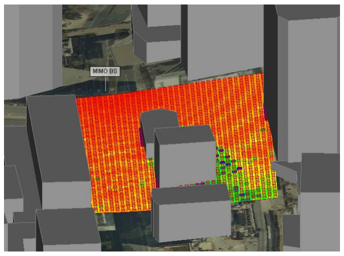


Fig. 5. Visualization of the urban outdoor scenario "Rosslyn, VA." generated with REMCOM's Wireless InSite ray-tracer.

urban outdoor scenario "Rosslyn, VA." The scenario is depicted in Fig. 5 and uses a dual-polarized uniform linear array (ULA) at the BS with a grid of possible UE positions separated by 2.5 m. For every generated MU-MIMO-OFDM channel matrix, we randomly select $U = 4$ UEs from the predefined grid.

Fig. 4(c) shows simulation results where we apply a total number of $N_{\text{MP}} = 12$ MP decoding iterations, which we evenly interleave with I detector stages. Remember that the total number of MP iterations equals the number of decoding iterations after each detection stage, i.e., $N_{\text{MP}} = \sum_{i=1}^{I} N_I$. In classical IDD, this situation would correspond to I IDD iterations, each of which applies $N_i = N_{\text{MP}}/I$ decoding iterations. The solid red curve in Fig. 4(c) corresponds to a non-IDD ($I = 1$) baseline with LMMSE detection followed by 12 LDPC decoding iterations. The solid orange curve corresponds to conventional IDD with $I = 2$. The solid blue and solid green lines correspond to DUIDD with LoCo-PIC and MMSE-PIC, respectively, both using $I = 2$. We see that DUIDD with MMSE-PIC performs 1.2 dB better than conventional IDD and 2.8 dB better than the non-iterative receiver at BLER $= 10^{-2}$. DUIDD with LoCo-PIC achieves roughly the same performance as IDD, but at lower complexity, as shown in Sec. V-D.

In Fig. 4(d), we analyze the detection and decoding convergence speed benefits from DUIDD. We compare IDD with a constant number of LDPC MP decoding iterations after each detection stage ($N_i = 12$), with $I = 2$ DUIDD with a total number of MP iterations that remains *constant* ($N_{\text{MP}} = 12$, i.e., $N_i = 4$). We see that DUIDD with LoCo-PIC and fewer LDPC decoding iterations achieves only slightly worse performance than IDD with MMSE-PIC running for $\{I = 2, N_i = 12\}$ iterations. Even more surprisingly, $\{I = 2, N_i = 6\}$ DUIDD with MMSE-PIC achieves better performance than IDD with $\{I = 4, N_i = 12\}$. This highlights that DUIDD can significantly improve the BLER performance and convergence speed, and thereby allows decreasing the computational complexity.

D. Performance vs. Complexity Trade-Off

We now compare the BLER performance versus the computational complexity. Since all of the considered methods apply the same channel estimation, noise-whitening, and LDPC decoding algorithms (with $N_{\text{MP}} = 12$), we focus on the complexity of the data detector. We count the number of real-valued multiplications $\#_{\text{MUL}}$ when implementing the detector with textbook algorithms for $T_D = 10$ coherent channels. Depending on what is optimal for each scenario and method, we consider either a Cholesky, QR, or LU decomposition for matrix inversion, and explicit calculation of the filter matrices or forward- and backward-substitution for equalization.

TABLE I
COMPUTATIONAL COMPLEXITY OF DATA DETECTION MEASURED IN 10^3 REAL-VALUED MULTIPLICATIONS FOR A COHERENCE OF $T_D = 10$

$B \times U$	$I = 1$ LMMSE	$I = 2$ LoCo-PIC	$I = 3$ LoCo-PIC	$I = 2$ MMSE-PIC	$I = 3$ MMSE-PIC
8×4	2.32	4.16	5.78	7.47	12.4
16×4	3.99	5.76	7.38	9.07	14.0
32×16	53.8	81.9	106	255	454

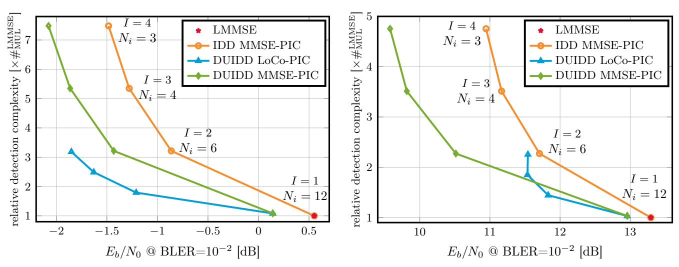
In Table I, we compare the complexity of non-iterative LMMSE detection, as well as LoCo-PIC and MMSE-PIC detection, for different numbers of BS antennas B, UEs U, and IDD iterations I. We can see, compared to LoCo-PIC, the complexity of MMSE-PIC is much larger and also growing faster with I and $B \times U$, because MMSE-PIC explicitly computes a matrix inverse for each coherent channel $t \in \{1, \ldots, T_D\}$ and detection stage i [19]. On the other hand, LoCo-PIC can reuse the filter matrix (once computed) for all t and i, and the complexity-increase mainly results from soft-symbol computation, PIC, equalization, and LLR de-mapping.

Fig. 6(a) compares the SNR performance measured in the minimum E_b/N_0 to achieve a BLER of 1% versus the computational complexity, normalized to the complexity of non-iterative LMMSE detection, for the Rayleigh fading scenario with perfect CSI from Sec. V-B. We see that by increasing the number of iterations I, the complexity increases and the SNR performance improves. While the DUIDD MMSE-PIC receiver has the same complexity as classical IDD, DUIDD achieves superior SNR performance, with gains of up to 0.6 dB. In contrast, DUIDD with LoCo-PIC exhibits significantly lower complexity while achieving almost the same performance as as the MMSE-PIC-based DUIDD receiver.

Fig. 6(b) compares the SNR performance versus the computational complexity for the ray-tracing scenario with estimated CSI from Sec. V-C. We observe even larger SNR performance gains for DUIDD with MMSE-PIC over classical IDD, with gains of up to 1.4 dB. DUIDD with LoCo-PIC is only Pareto optimal for $I = 2$ and outperformed by DUIDD with MMSE-PIC at higher complexity.

VI. CONCLUSIONS

We have proposed deep-unfolded interleaved detection and decoding (DUIDD), a novel paradigm that outperforms traditional iterative detection and decoding (IDD) receivers in terms of SNR performance and complexity. The key idea is to break the strict separation between data detector and channel decoder in a traditional IDD receiver and to jointly optimize all algorithm parameters using deep unfolding and ML. To further improve the efficacy of DUIDD, we have introduced new hyperparameters that enable optimizing soft-information exchange, message damping, and state forwarding. To reduce the complexity of SISO data detection, we proposed



(a) (b)

Fig. 6. Computational complexity vs. performance (measured in minimal SNR required to achieve BLER $= 1\%$) trade-off for a fixed number $N_{\mathrm{MP}} = 12$ of total MP decoding iterations. The subfigures consider a MU-MIMO-OFDM system (a) with $B \times U = 8 \times 4$ i.i.d. Rayleigh fading channels and perfect CSI, and (b) with $B \times U = 16 \times 4$ ray-tracing channels and estimated CSI.

a low-complexity variant of the SISO MMSE-PIC algorithm. In simulations with NVIDIA's Sionna [16], our 5G-near MU-MIMO-OFDM DUIDD receivers have shown SNR performance gains of up to 1.4 dB and reduced complexity compared to IDD.

There are many avenues for future work. An automated search for the optimal interleaving pattern of the detector and decoder stages given a complexity constraint is ongoing. Furthermore, an investigation of other data detectors and channel decoders within the DUIDD framework may further improve the performance-complexity trade-off.

References

[1] European Telecommunications Standards Institute, "5G NR multiplexing and channel coding," Apr. 2021, ETSI 3GPP TS 38.212 version 16.5.0 Release 16.

[2] J. Hoydis, F. A. Aoudia, A. Valcarce, and H. Viswanathan, "Toward a 6G AI-native air interface," *IEEE Commun. Mag.*, vol. 59, no. 5, pp. 76–81, May 2021.

[3] D. Seethaler, G. Matz, and F. Hlawatsch, "An efficient MMSE-based demodulator for MIMO bit-interleaved coded modulation," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*. IEEE, Nov. 2004.

[4] A. Burg, S. Haene, D. Perels, P. Luethi, N. Felber, and W. Fichtner, "Algorithm and VLSI architecture for linear MMSE detection in MIMO-OFDM systems," in *IEEE Int. Symp. Circuits and Syst. (ISCAS)*, May 2006.

[5] J. Ketonen, M. Juntti, and J. R. Cavallaro, "Performance—complexity comparison of receivers for a LTE MIMO–OFDM system," *IEEE Trans. Signal Process.*, vol. 58, no. 6, pp. 3360–3372, Jun. 2010.

[6] M. Wu, B. Yin, G. Wang, C. Dick, J. Cavallaro, and C. Studer, "Large-scale MIMO detection for 3GPP LTE: Algorithms and FPGA implementations," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 916–929, Oct. 2014.

[7] L. Lu, G. Y. Li, A. L. Swindlehurst, A. Ashikhmin, and R. Zhang, "An overview of massive MIMO: Benefits and challenges," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 742–758, Oct. 2014.

[8] S. Yang and L. Hanzo, "Fifty years of MIMO detection: The road to large-scale MIMOs," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 1941–1988, Sep. 2015.

[9] G. Peng, L. Liu, S. Zhou, S. Yin, and S. Wei, "A 1.58 Gbps/w 0.40 Gbps/mm2 ASIC implementation of MMSE detection for 128 × 8 64-QAM massive MIMO in 65 nm CMOS," *IEEE Trans. Circuits Syst. I*, vol. 65, no. 5, pp. 1717–1730, May 2018.

[10] B. Hochwald and S. Ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Trans. Commun.*, vol. 51, no. 3, pp. 389–399, Mar. 2003.

[11] C. Studer, "Iterative MIMO decoding: Algorithms and VLSI implementation aspects," Ph.D. dissertation, ETH Zurich, Jul. 2009.

[12] N. Preyss, A. Burg, and C. Studer, "Layered detection and decoding in MIMO wireless systems," in *Proc. of Conf. on Design and Archit. for Signal and Image Process. (DASIP)*. IEEE, Oct. 2012, pp. 1–8.

[13] W.-C. Sun, W.-H. Wu, C.-H. Yang, and Y.-L. Ueng, "An iterative detection and decoding receiver for LDPC-coded MIMO systems," *IEEE Trans. Circuits Syst. I*, vol. 62, no. 10, pp. 2512–2522, Oct. 2015.

[14] M. Wu, C. Dick, J. R. Cavallaro, and C. Studer, "Iterative detection and decoding in 3GPP LTE-based massive MIMO systems," in *Proc. Euro. Sig. Proc. Conf. (EUSIPCO)*, Sep. 2014, pp. 96–100.

[15] A. Balatsoukas-Stimming and C. Studer, "Deep unfolding for communications systems: A survey and some new directions," in *IEEE Int'l Workshop on Signal Process. Sys. (SiPS)*. IEEE, Oct. 2019.

[16] J. Hoydis, S. Cammerer, F. Ait Aoudia, A. Vem, N. Binder, G. Marcus, and A. Keller, "Sionna: An open-source library for next-generation physical layer research," *arXiv preprint*, Mar. 2022.

[17] R. Wiesmayr, G. Marti, C. Dick, H. Song, and C. Studer, "Bit error and block error rate training for ML-assisted communication," *arXiv:2210.14103*, Oct. 2022.

[18] M. Tüchler, A. Singer, and R. Koetter, "Minimum mean squared error equalization using a priori information," *IEEE Trans. Signal Process.*, vol. 50, no. 3, pp. 673–683, Mar. 2002.

[19] C. Studer, S. Fateh, and D. Seethaler, "ASIC implementation of soft-input soft-output MIMO detection using MMSE parallel interference cancellation," *IEEE J. Solid-State Circuits*, vol. 46, no. 7, pp. 1754–1765, Jul. 2011.

[20] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be'ery, "Deep learning methods for improved decoding of linear codes," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 119–131, Jan. 2018.

[21] M. Lian, C. Hager, and H. D. Pfister, "What can machine learning teach us about communications?" in *Proc. IEEE Inf. Theory Workshop (ITW)*. IEEE, Nov. 2018.

[22] X. Lin, "An overview of 5G advanced evolution in 3GPP release 18," *arXiv:2201.01358*, Jan. 2022.

[23] T. J. O'Shea, T. Erpek, and T. C. Clancy, "Deep learning based MIMO communications," *arXiv:1707.07980*, Jul. 2017.

[24] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Trans. Signal Process.*, vol. 67, no. 10, pp. 2554–2564, May 2019.

[25] M. Khani, M. Alizadeh, J. Hoydis, and P. Fleming, "Adaptive neural signal detection for massive MIMO," *IEEE Trans. Wireless Commun.*, vol. 19, no. 8, pp. 5635–5648, Aug. 2020.

[26] S. Cammerer, J. Hoydis, F. A. Aoudia, and A. Keller, "Graph neural networks for channel decoding," *arXiv:2207.14742*, Jul. 2022.

[27] S. Dörner, S. Cammerer, J. Hoydis, and S. ten Brink, "Deep learning based communication over the air," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 132–143, Feb. 2018.

[28] F. A. Aoudia and J. Hoydis, "Model-free training of end-to-end communication systems," *IEEE J. Sel. Topics Signal Process.*, vol. 37, no. 11, pp. 2503–2516, Nov. 2019.

[29] J. Song, C. Häger, J. Schröder, T. J. O'Shea, E. Agrell, and H. Wymeersch, "Benchmarking and interpreting end-to-end learning of MIMO and multi-user communication," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, pp. 7287–7298, Sep. 2022.

[30] S. Cammerer, F. A. Aoudia, S. Dorner, M. Stark, J. Hoydis, and S. ten Brink, "Trainable communication systems: Concepts and prototype," *IEEE Trans. Commun.*, vol. 68, no. 9, pp. 5489–5503, Sep. 2020.

[31] J. Zhang, C.-K. Wen, and S. Jin, "Adaptive MIMO detector based on hypernetwork: Design, simulation, and experimental test," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 65–81, Jan. 2022.

[32] L. V. Nguyen, N. T. Nguyen, N. H. Tran, M. Juntti, A. L. Swindlehurst, and D. H. N. Nguyen, "Leveraging deep neural networks for massive MIMO data detection," *IEEE Wireless Commun.*, pp. 1–7, May 2022.

[33] M. Witzke, S. Baro, F. Schreckenbach, and J. Hagenauer, "Iterative detection of MIMO signals with linear detectors," in *Asilomar Conf. Signals, Syst., Comput.* IEEE, Nov. 2002.

[34] J. Vogt and A. Finger, "Improving the max-log-MAP turbo decoder," *Electron. Lett.*, vol. 36, no. 23, p. 1, Nov. 2000.

[35] A. Tomasoni, M. Ferrari, D. Gatti, F. Osnato, and S. Bellini, "A low complexity turbo MMSE receiver for W-LAN MIMO systems," in *Proc. IEEE Int. Conf. Commun. (ICC)*. IEEE, Jun. 2006.

[36] I. Collings, M. Butler, and M. McKay, "Low complexity receiver design for MIMO bit-interleaved coded modulation," in *Proc. IEEE Int'l Symp. Spread Spectrum Techniques and Appl.* IEEE, Aug. 2004.

[37] D. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[38] V. Savin, "Gradient descent bit-flipping decoding with momentum," in *Int. Symp. on Topics in Coding (ISTC)*. IEEE, Aug. 2021.